AD-A047 011 NAVAL SURFACE WEAPONS CENTER WHITE OAK LAB SILVER SP--ETC F/G 9/2
GENERALIZED MULTI-CHANNEL ANALYSIS PROGRAMS FOR THE DART (DIGIT--ETC(U)
FEB 77 C A SHIVELY

UNCLASSIFIED NSWC/WOL-TR-77-3 NL

| OF |
AD
A047 011

END
DATE
FILMED
1-78
DDC

(14) NSWC/WOL-TR-77-3

(6) GENERALIZED MULTI-CHANNEL ANALYSIS PROGRAMS FOR THE DART (DIGITAL ANALYSIS IN REAL TIME) COMPUTER SYSTEM.

(10) BY CURTIS A. SHIVELY (9) Final rept.,

ORDNANCE SYSTEMS DEVELOPMENT DEPARTMENT

(11) 2 FEB [redacted]977 (12) 50 p. (16) W21W9

Approved for public release; distribution unlimited.

NAVAL SURFACE WEAPONS CENTER

Dahlgren, Virginia 22448 • Silver Spring, Maryland 20910

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NSWC/WOL/TR 77-3
2. GOVT ACCESSION NO.:
3. RECIPIENT'S CATALOG NUMBER:

4. TITLE *(and Subtitle)*: GENERALIZED MULTI-CHANNEL SPECTRUM ANALYSIS PROGRAMS FOR THE DART (Digital Analysis in Real Time) COMPUTER SYSTEM

5. TYPE OF REPORT & PERIOD COVERED: Final

6. PERFORMING ORG. REPORT NUMBER:

7. AUTHOR(s): Curtis A. Shively

8. CONTRACT OR GRANT NUMBER(s):

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Naval Surface Weapons Center
White Oak Laboratory
Silver Spring, MD 20910

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
63259N, W21W9, WU6130

11. CONTROLLING OFFICE NAME AND ADDRESS:
Naval Air Systems Command
Washington, DC

12. REPORT DATE: 2 February 1977

13. NUMBER OF PAGES: approx. 40

14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*:

15. SECURITY CLASS. *(of this report)*: UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*
real time signal processing
spectrum analysis
FFT

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This report describes software programs for a system of NOVA computers to accumulate multichannel input time samples and generate their Fourier transform coefficients in real time. Also described are programs to receive the DFT coefficients for further processing under a BASIC interpreter. Program Listings and comments on usage are included.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

SUMMARY

This report describes software written to use the DART computer system for real time narrow band spectrum analysis. This material is of general interest to those engaged in programming minicomputer systems for signal processing applications, and of particular interest to those using the DART system for investigating frequency domain signal processing techniques. This work was performed in the Digital and Signal Processing Branch of the Ordnance Systems Development Department and was funded by Naval Air Systems Command.

Edward C. Whitman
EDWARD C. WHITMAN
By direction

CONTENTS

APPENDICES

FIGURES

GENERALIZED MULTI-CHANNEL ANALYSIS PROGRAMS FOR THE DART (Digital Analysis in Real Time) COMPUTER SYSTEM

I. INTRODUCTION

Many techniques for processing signals involve narrow band spectrum analysis as a first step. The computer programs described herein were written to use a system of minicomputers to perform spectrum analysis on a group of analog input channels. The resulting discrete Fourier transform (DFT) (reference 1) coefficients are periodically sent to other computers in the system for further processing in real time.

Figure 1 shows a block diagram of the DART (digital analysis in real time) minicomputer system developed by the Signal and Digital Processing Branch. Four Data General NOVA 800 minicomputers (reference 2) perform the computations. A multiprocessor communications adapter (MCA) provides for transfer of data among the computers. Multichannel A/D and D/A converters are interfaced to CPU-B. CPU-A hosts a high speed special purpose hardware fast Fourier transform (FFT) computer (reference 3). The remaining two computers C and D are each interfaced to a graphics terminal, disc memory, and digital magnetic tape unit.

The channels of data to be analyzed are fed into the A/D converters on CPU-B where they are periodically sampled and digitized. A group of these time samples is accumulated in the core memory of CPU-B for every input channel. Each block of time samples is transferred from CPU-B via the MCA to CPU-A for computation of the corresponding DFT coefficients. A selected frequency band of coefficients is made available to be sent from CPU-A to computer C or D for further processing.

The software programs performing the above functions in computers A and B are written in NOVA assembly language for maximum efficiency of memory and execution speed. With program PAPB (Appendix A) in CPU-B and PAPABFP (Appendix B) in CPU-A, these two computers may function independently as a multi-channel spectrum analysis system, with display of a single channel

---

[1] W. T. Cochran, J. W. Cooley, et al., "What is the Fast Fourier Transform?" Proceedings of the IEE, Vol. 55, pg. 1664-1674, October, 1967.

[2] How to Use the NOVA Computers, Data General Corporation, Southboro, Mass., 1971.

[3] Operating Manual for System 306/400 Spectrum Analyzer, Elsytec Corporation, Syossett, New York, 1972.

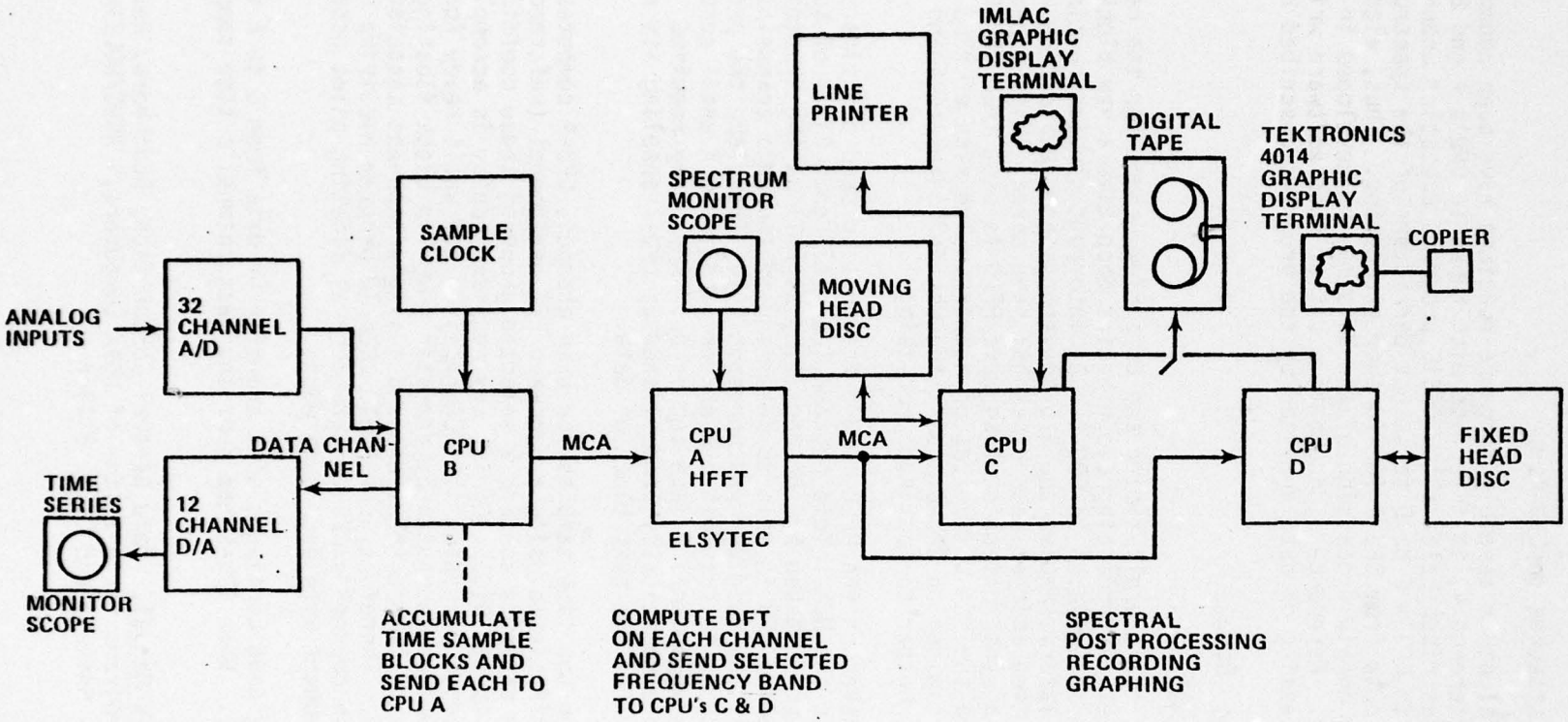

FIG. 1 DART SYSTEM BLOCK DIAGRAM

on the monitor scope attached to CPU-A.

However, several other assembly language routines have been added to a NOVA BASIC interpreter (reference 4) to interface with those in CPU's A and B. These CALL routines described below allow BASIC programs executing under an interpreter resident in CPU's C or D to select parameters of the spectrum analysis and receive data from CPU-A for further processing. Thus, algorithms for further frequency domain processing of the data may be developed in much higher level language. Reference 5 documents a set of such software written to perform array processing on data analyzed by the programs described herein.

II. System Processing Scheme

A. Processing Cycle

Figure 2 shows the overall timing and data flow scheme for the real time spectral analysis. A processing cycle begins each time a new block of time samples has been accumulated in CPU-B. An interrupt service routine (described in detail later) samples the A/D converters and sets a flag word XCFLG in memory each time another data block has been saved. When a new data block is ready, a synchronization flag word PFLG is sent from CPU-B to A and D(C) in turn via the MCA communication network. The sign of PFLG is selected by a console switch on CPU-B and it thereby indicates to A and the destination computer whether to continue processing.

Following transfer of the PFLG, CPU-B attempts to send the time samples for channel 1 to A via the MCA. This network (see reference 2) is of such a nature that action must be taken by the receiving as well as the transmitting CPU. Moreover, a sending CPU is able to determine if the data transmission was accomplished or if the receiver was unresponsive. Therefore the program in CPU-B repeatedly tries to transmit the channel 1 data to A until successful Then it attempts to send channel 2 data and so on. Any delay required between transmissions is thus automatically accomplished by CPU-A enabling its receiver only when it is ready for the next block of data.

After receiving the time samples for each channel, CPU-A computes the corresponding DFT coefficients using a hardware FFT peripheral (reference 3). The program in A then attempts to send a selected group of these coefficients to the destination computer D(C). Again, any necessary delay is accomplished by the destination computer keeping its MCA receiver off until ready for more data. The DFT coefficients computed by computer A are in block floating point format, i.e., a group of 16-bit fractions and a single exponent associated with the entire block. Therefore, the BASIC CALL 12 program receiving data in CPU-D(C) converts each channel's DFT to Data General floating point notation having a separate exponent with each data word.

The processing thus continues with transfer of data from B to A to D(C) one channel at a time. When CPU-B has sent the last channel's time samples

[4] Extended BASIC User's Manual, Data General Corporation, Southboro, Mass., 1972.
[5] "Array Processing Programs for a System of NOVA Computers," NSWC/WOL/TR 77-2, January, 1977, C. A. Shively and L. S. Biller.
[2] Ibid.
[3] Ibid.

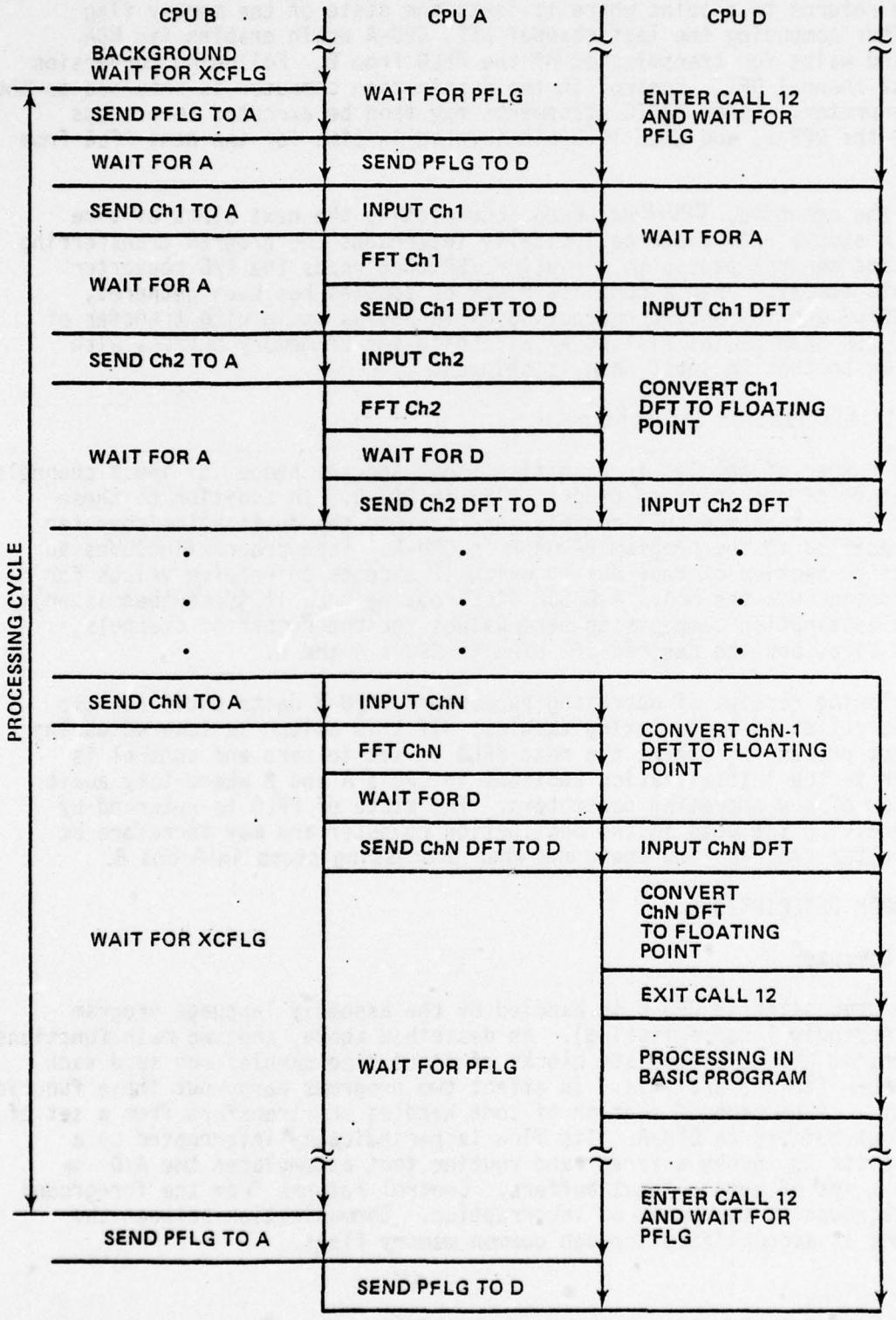

FIG. 2 DATA FLOW AND TIMING

to A, PAPB returns to a point where it tests the state of the memory flag XCFLG. After computing the last channel DFT, CPU-A again enables its MCA receiver and waits for transmission of the PFLG from B. Following conversion of the last channel DFT, control in the destination computer is returned to the BASIC interpreter. Other BASIC statements may then be executed to process or display the DFT's, and CALL 12 again invoked in time for the next PFLG from CPU-A.

In the meantime, CPU-B has been accumulating the next block of time samples. A sample rate clock periodically interrupts the program transferring data to A and control passes to a routine ISR that reads the A/D converter samples into memory. When a complete block of samples has been gathered, ISR sets SCFLG and the entire processing cycle begins again with transfer of the PFLG. ISR then begins filling an alternate set of memory buffers with time samples so that no input data is skipped.

B. Initialization and Control

The number of samples in each time block and the number of input channels are parameters of operation of program PAPB in CPU-B. In addition to these numbers, the exact band of DFT coefficients sent to the destination computer must be specified to the program PAPABFP in CPU-A. Each program includes an initialization section of code during which it expects to receive values for these parameters via the MCA. A BASIC CALL routine CALL 11 (described later) allows the destination computer to send values for the number of channels, data block size, and the desired DFT band to CPU's A and B.

Following receipt of operating parameters, CPU-B waits until console switch Ø is raised to begin taking samples. If this switch is lowered during a subsequent processing cycle, the next PFLG is set to zero and control is passed back to the initialization sections in CPU's A and B where they await transmission of new operating parameters. The state of PFLG is returned by CALL 12 receiving the data in the destination computer and may therefore be tested to alter program flow where and when processing stops in A and B.

III. PROGRAM DESCRIPTIONS

A. Computer B

The processing in CPU-B is handled by the assembly language program PAPB (see Appendix A for a listing). As described above, the two main functions of this program are to accumulate blocks of input time samples and send each block to CPU-A for DFT analysis. In effect two programs carry out these functions concurrently. A background section of code handles the transfers from a set of memory output buffers to CPU-A. Its flow is periodically interrupted by a real time clock to invoke a foreground routine that accumulates the A/D samples in a set of memory input buffers. Control returns from the foreground to the background at the point of interruption. Communication between the two routines is accomplished through common memory flags.

Background

A flow diagram of the background section of PAPB is shown in Figure 3. Upon initial entry, the background waits to receive values for the number of channels and time block size from the destination computer via the MCA. PAPB then allocates the corresponding memory input and output buffers and halts if insufficient memory is available in computer B. After console switch Ø is raised, the clock interrupt is enabled to permit periodic entry into the foreground program.

A processing cycle begins when the foreground sets memory flag XCFLG to indicate that a new block of time samples is ready for transmission to CPU-A. If XCFLG is already set when the background first tests it, the program halts to indicate that it cannot process contiguous time blocks at the given sample rate and number of channels. Similarly, if A is not responsive to receiving the subsequent PFLG, PAPB halts to warn the user that the processing cycle in CPU-A is too time consuming.

If console switch Ø is down, control then returns to the background initialization at PAPB. Otherwise, the background program reads the console switches and sets an address word in the foreground to output samples from the correspondingly numbered input channel. Then the program sends each channel's block of time samples to CPU-A in order and returns to test XCFLG for start of the next cycle.

Foreground

Upon occurrence of an interrupt from the real time clock (reference 2) control is passed to the interrupt service routine ISR shown in Figure 4. ISR sets the A/D converter to sample all analog inputs and then store the sample from each channel in turn into a memory buffer. The sample from the channel selected for display is then output to the D/A converter. Reference 6 gives a complete description of the operation of the A/D and D/A converters interfaced to CPU-B.

ISR next concatenates the new sample from each channel onto a time series block being formed in memory. If the sample count SCNT is odd, each 8-bit sample is packed into the right byte of a memory word. If SCNT is even, packing is into the left byte. If the newly incremented sample count is equal to the block size, SCNT is reset to Ø, XCFLG is set to -1, and the input and output buffer pointers are swapped. Control is then returned to the background program at the point of interruption.

B. Computer A

A better understanding of the software performing spectrum analysis in computer A will result by first looking at its hardware configuration. CPU-A is the central processor of a stand-alone spectrum analyzer built commercially by the Elsytec Corporation (reference 3). A special purpose peripheral computes

---

[3]Ibid.
[2]Ibid.
[6]"Naval Ordnance Laboratory Acoustic Signal Generator System," Systems Technology Assos. Inc., Falls Church, VA, 1973.

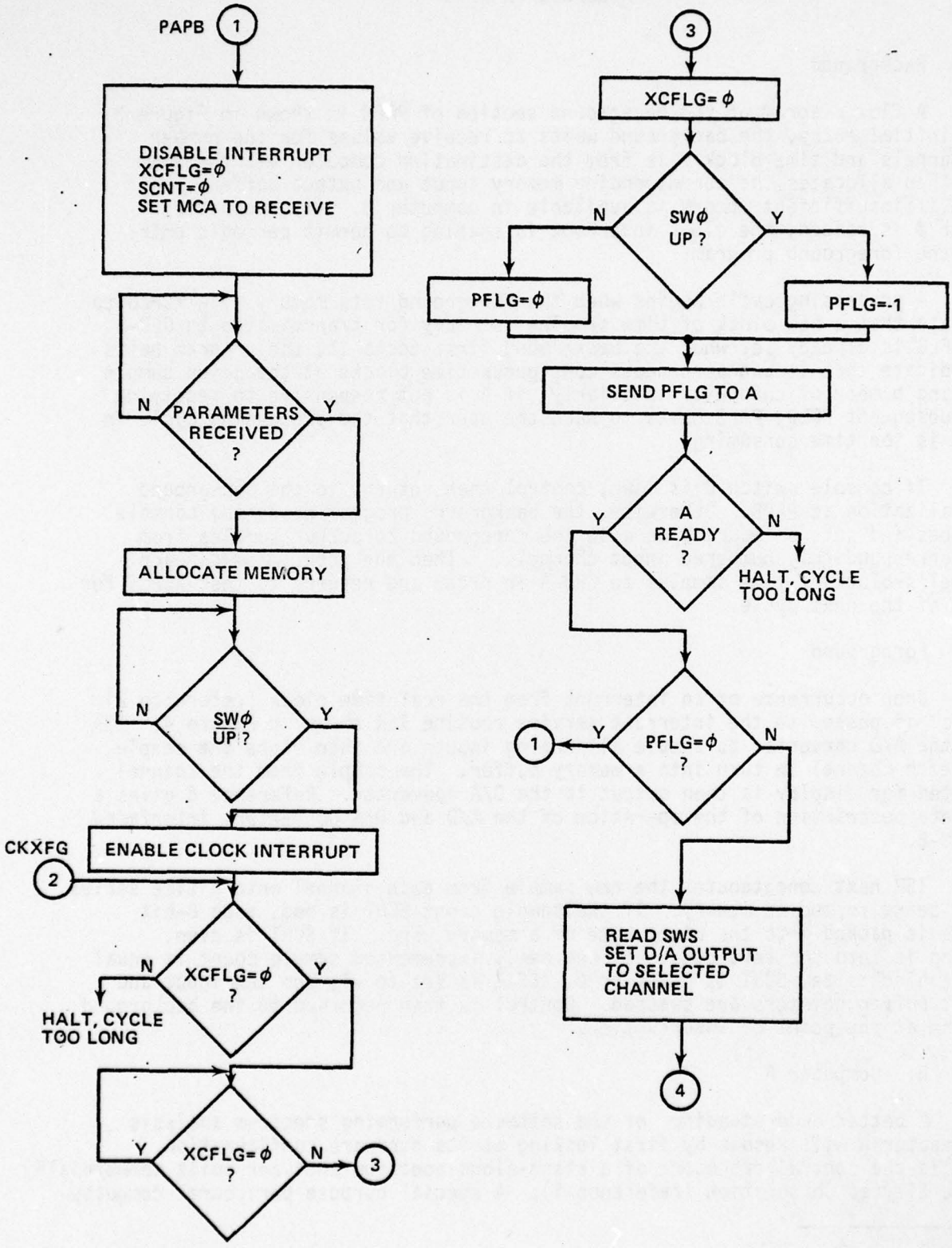

FIG. 3-A BACKGROUND FLOW OF PAPB

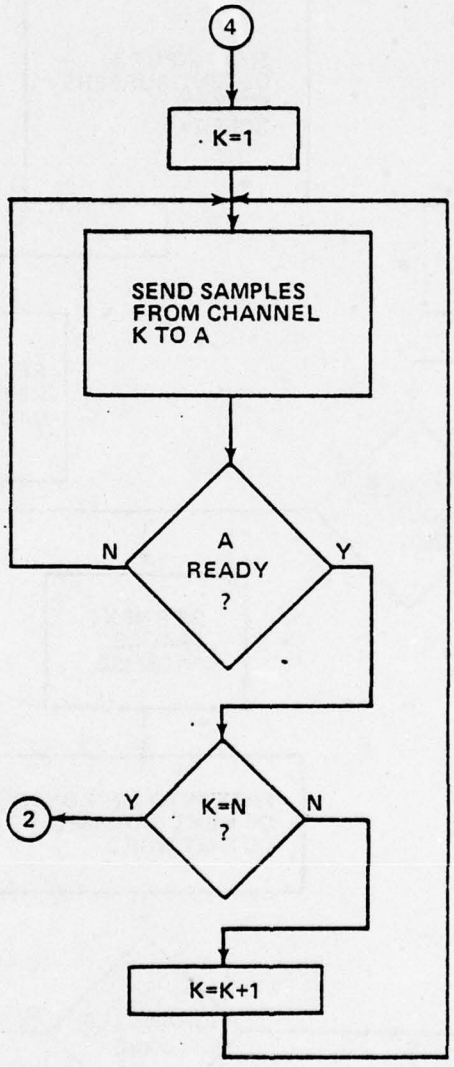

FIG. 3-B BACKGROUND FLOW OF PAPB

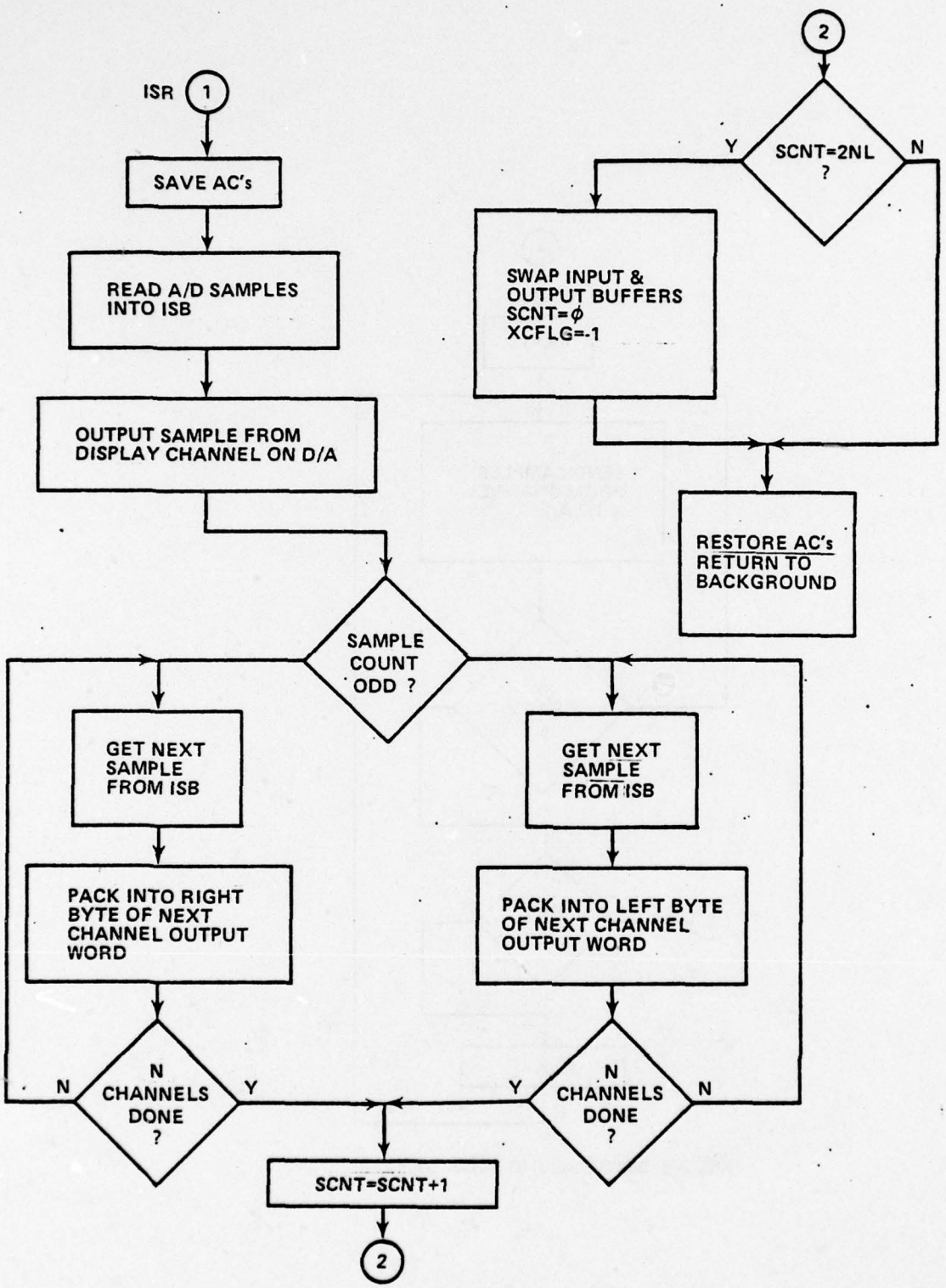

FIG. 4 FOREGROUND FLOW OF PAPB

the discrete Fourier transform of data in the memory of computer A using a Fast Fourier Transform (FFT) algorithm (reference 1). Analyzer peripheral hardware also includes a control panel and output buffers for refreshing an oscilloscope display.

Software provided by the manufacturer gives a variety of operations on one or two input channels. A new main program PAPABFP has been written to use computer A for multi-channel spectrum analysis in conjunction with the other computers in the DART system. A separate subroutine CNTF (Appendix C) was written to normalize DFT results. Elsytec proprietary subroutines are still used for some of their original functions. HFFTI and CALLS control the operation of the hardware FFT peripheral, SCPLD handles display of data on the monitor scope, and an auxiliary math routine MATHS finds the largest number in a data array.

A flow diagram of PAPABFP is shown in Figure 5 and an assembly listing is given in Appendix B. Unlike PAPB in which both foreground and background programs run interlaced in time, PAPABFP is straight forward execution of a single process. As in PAPB, PAPABFP initially waits for parameters via the MCA. These include the number of channels NCH, number of lines in FFT NL, number of lines returned to destination computer NLR, and the cell number of the first line returned RLD.

Following reception of parameters CPU-A enters the main processing loop where it waits to receive the PFLG from CPU-B. If CPU-A is unsuccessful in relaying PFLG on to the destination computer, it sets a flag MTCRF to skip sending DFT data computer later in the cycle. Thus, once initialized, PAPB and PAPBFP may perform spectrum analysis regardless of whether a post processing computer is receptive. If used, the destination computer may be either CPU C or D as selected by the MCA code word TMMC in PAPABFP.

If PFLG = Ø, control returns to the initialization of parameters via the MCA. Otherwise, the console switches are read to determine which channel spectrum will be displayed later on the oscilloscope. The MCA receiver is then enabled to await transmission of channel 1 time samples from CPU-B.

After each channel's samples are received and unpacked, the hardware FFT peripheral (HFT) is invoked via Elsytec proprietory subroutines CALLS and HFFTI to compute their DFT. The HFT is about an order of magnitude faster than software, since it converts 1024 real time points into 512 complex frequency coefficients in about 18 milliseconds. The HFT automatically scales the data when necessary to avoid overflow in the fixed point computations. The resulting output DFT coefficients are in block floating point notation i.e., a group of 16-bit binary fractions all associated with the same base two exponent.

---

[1] Ibid.

If the destination computer accepted the PFLG, the group of NLR DFT cells starting at cell RLD are moved to a different memory buffer for output. A block normalization is then performed so that their largest mantissa is between 1/2 and 1. CPU-A then repeatedly attempts to send these DFT's to the destination computer until it is ready to receive them.

If the current channel being processed was selected for display, its entire band of DFT's are converted to polar coordinates. An Elsytec proprietary subroutine SCPLD is then invoked to output the spectrum to the scope refresh buffer and to select the display format. If console switch Ø is up, the display is logarithmic, down, linear. Each of the two output buffers holds a maximum of 512 frequency lines. Therefore, if a time sample block size of more than 2048 is used, only the first 1024 DFT coefficients will be displayed.

C. Destination Computer

An assembly language routine RMDFE has been added to a NOVA BASIC interpreter (reference 4) to allow BASIC programs to interface with PAPB and PAPABFP. The assembly listing in Appendix D includes extensive comments on RMDFEF and its usage. One section of RMDFEF allows the destination computer to initialize PAPB and PAPABFP, and the other to receive the PFLG and DFT data from CPU-A.

Initialization is achieved by invoking

```
CALL 11,NCH,NL,NLR, RLD
```

where NCH is the number of input channels, NL is the number of spectral lines resulting from DFT, NLR is the number of lines returned to the destination computer, and RLD is the cell number of the first coefficient returned.

A diagram of the flow of CALL 11 is shown in Figure 6. The program straight-forwardly attempts to send parameter words first to computer A and then to computer B via the MCA. Each transmission is tried repeatedly until successful. Therefore, control will not pass from CALL 11 back to the BASIC interpreter until computers A and B have accepted their parameters.

Reception of the DFT coefficients is invoked by

```
CALL 12,R(Ø),I(Ø),P
```

where R and I are arrays into which the real and imaginary parts of the complex DFTs are stored in order by channel. Arrays R and I should be dimensioned NCH*NLR elements. The state of PFLG received is passed to the BASIC program through variable P, which may be tested to determine whether computers A and B are continuing to process data.

---

[4] Ibid.

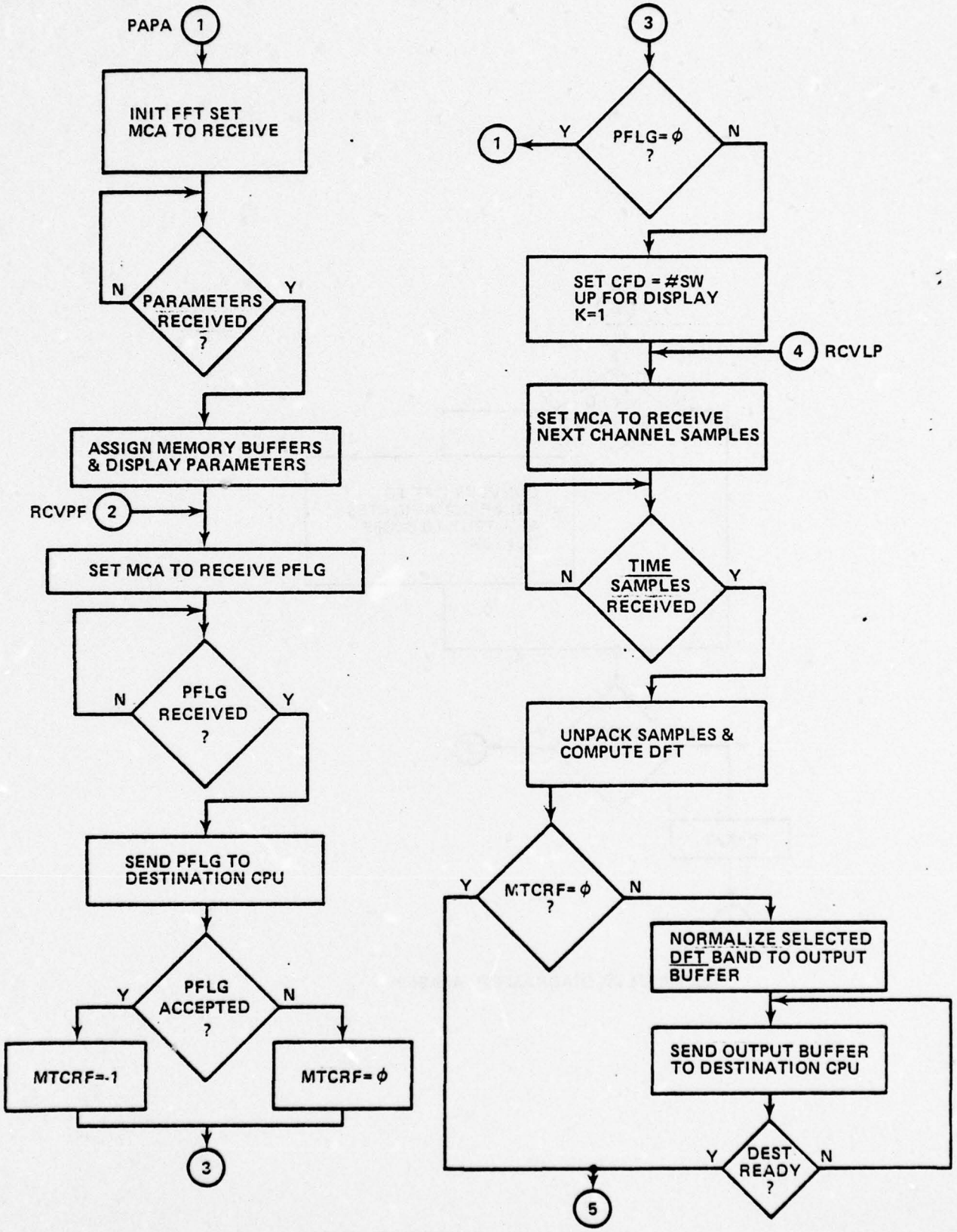

FIG. 5-A FLOW DIAGRAM OF PAPABFP

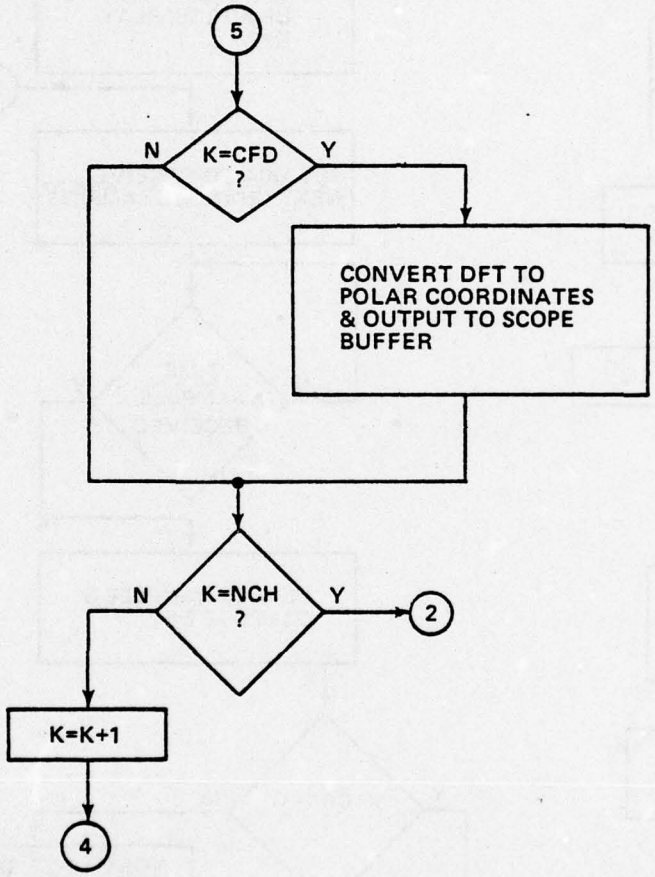

FIG. 5-B FLOW DIAGRAM OF PAPABFP

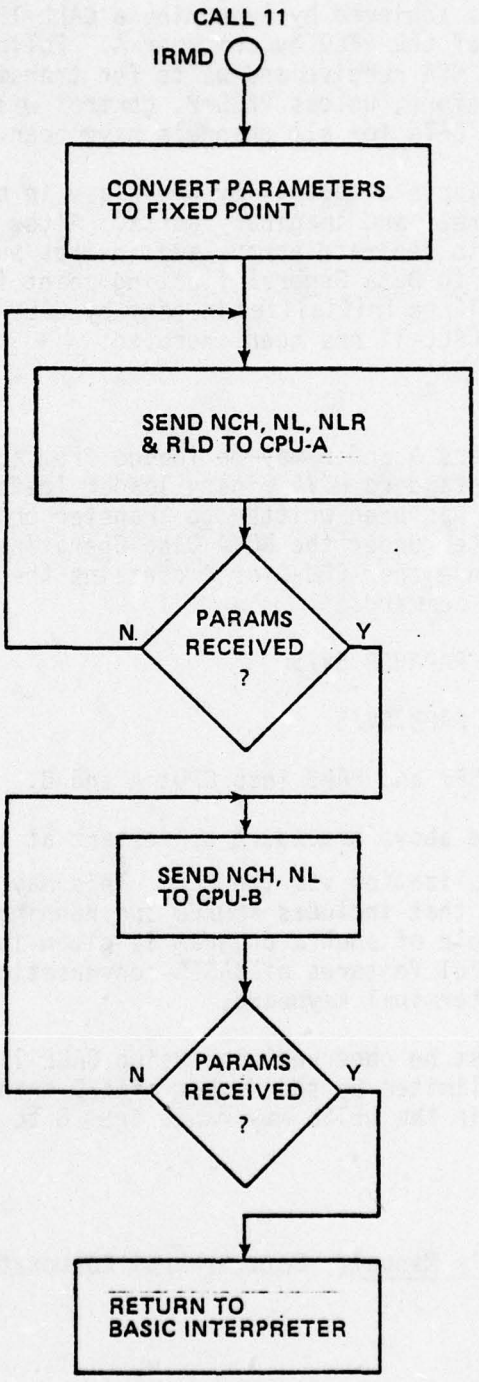

FIG. 6 FLOW DIAGRAM OF CALL 11

A flow diagram of CALL 12 is shown in Figure 7. Real time synchronization of this program with CPU-A is achieved by executing a CALL 12 each processing cycle prior to transmission of the PFLG by computer A. Following reception of PFLG, CALL 12 enables the MCA receiverand waits for transmission of each channel's DFT in turn. Therefore, unless PFLG=Ø, control will not return to BASIC from CALL 12 until the DFTs for all channels have been received.

The DFT complex coefficients computed by CPU-A are in block floating point format with alternate real and imaginary parts. After receiving them, RMDFEF moves the components to separate arrays and invokes subroutine BFLT (Appendix E) to convert them to Data General floating point format. It is also important to note that CALL 12 is initialized in part by CALL 11, and therefore may not be invoked unless a CALL 11 has been executed.

IV. Operating Procedures

The programs for computers A and B may be loaded from absolute binary object paper tape using the standard NOVA binary loader (reference 2). However, a program RLOAD (Appendix F) has been written to transfer programs from disc to the memory of an execution computer under the NOVA Disc Operating System (reference 7). If a disc operating system on either CPU-C or D contains the save files PAPB.SV, PAPABFP.SV and RLOAD.SV, the commands

```
RLOAD/A PAPABFP.SV/S

RLOAD/B PAPB.SV/S
```

may be executed to load PAPABFP and PAPB into CPUs A and B.

Following loading by the above procedure or restart at location 1000_8, these programs are ready for initialization via the MCA. This may be accomplished by invoking a BASIC interpreter that includes RMDFED and running a program that executes a CALL 11. An example of such a program is given in Appendix G. It illustrates one of the powerful features of BASIC-conversational input of program parameters from the terminal keyboard.

Several restrictions must be observed when using CALL 11. The number of channels NCH may be 1-32 as limited by the number of A/D channels operative on CPU-B. The number of lines in the DFTNL may range from 8 to 8192 in powers of two.

---

[2] Ibid.

[7] Disc Operating System User's Manual, Data General Corporation, Southboro, Mass., 1971.

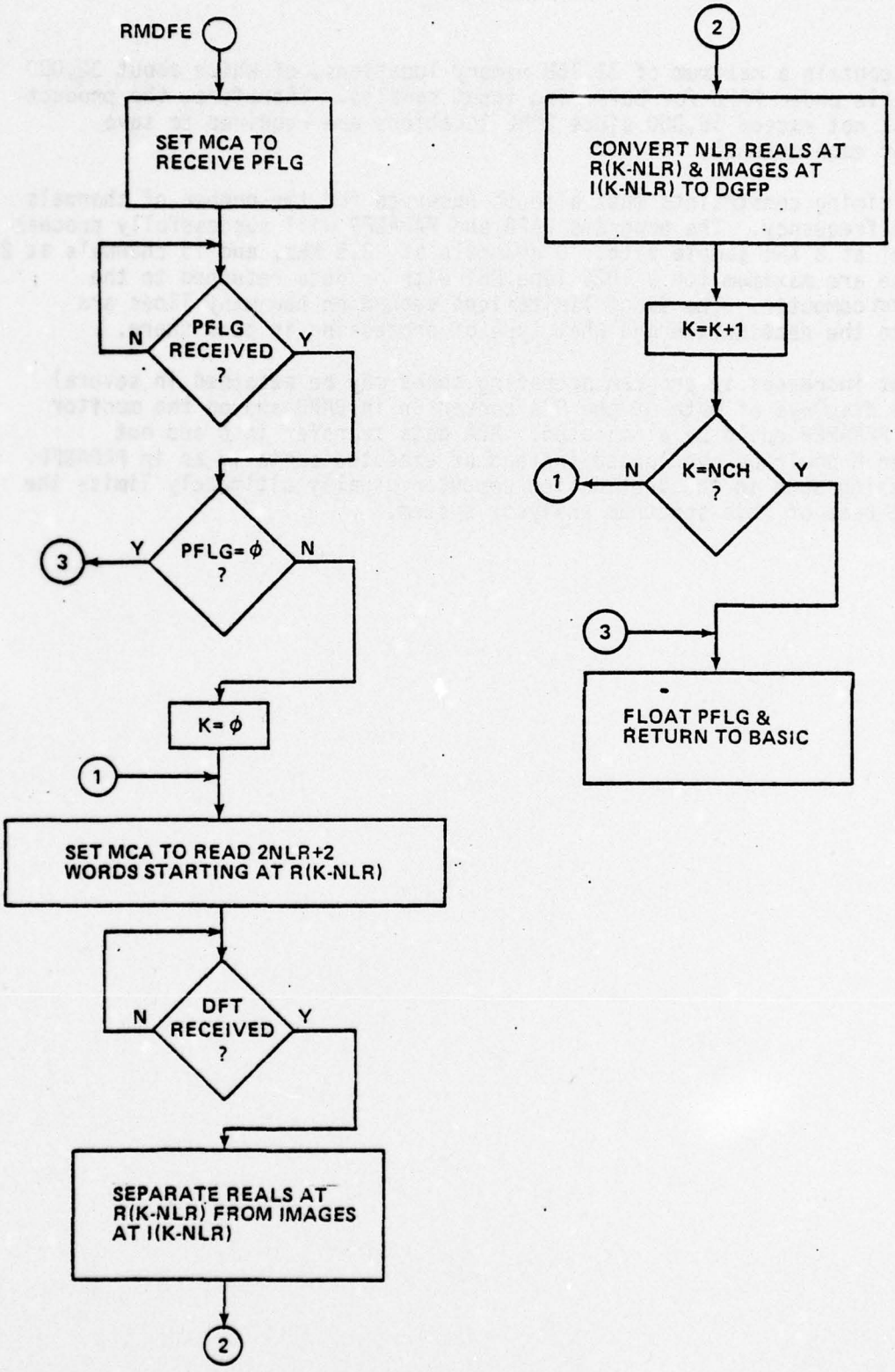

FIG. 7 FLOW DIAGRAM OF CALL 12

CPU-B can contain a maximum of 32,768 memory locations, of which about 32,000 are available under PAPB for buffering input samples. Therefore, the product ML*NCH must not exceed 16,000 since 2*NL locations are required to save samples for each channel.

Some timing constraints must also be observed for the number of channels and sample frequency. The programs PAPB and PAPABFP will successfully process one channel at 8 KHz sample rate, 10 channels at 3.5 KHz, and 15 channels at 2 kHz etc. These are maximum for a 1024 line DFT with no data returned to the destination computer. The exact limitations depend on how many lines are returned to the destination and what type of processing is done there.

Slight increases in program operating speed may be attained in several ways. The displays of data on the D/A converter in PAPB and on the monitor scope in PAPABFP could be eliminated. MCA data transfer into and out of computer A could be overlapped instead of executed serially as in PAPABFP. However, the processing done in the destination computer usually ultimately limits the throughput rate of this spectrum analysis system.

0001 PAPB

APPENDIX A - LISTING OF PAPB

```
                ; 10/25/73      C. SHIVELY
                ;               ARRAY PROCESSING
                ; PROGRAM FOR A/D MACHINE B
                        .TITL   PAPB
        000020          .LOC    20
00020 000000 OADP:      0
00021 000000 IADP:      0
00022 000000 ISDP:      0
                        .ZREL
00000-002000 BLL:       2000            ;BUFFER LOWER LIMIT
00001-060000 MUPL:      60000           ;1+HIGHEST MEMORY ADDRESS
00002-000000 XCFLG:     0               ;BEGIN XFER CYCLE FLAG
00003-177773 INTM:      177773
00004-000000 SCNT:      0               ;SAMPLE COUNT
00005-000007-PAAD:      NCH
00006-000002 NP:        2               ;# OF PARAMETERS
00007-000013 NCH:       11.             ;# OF CHANNELS
00010-002000 NL:        1024.           ;# FREQ LINES=HALF # SAMPLES/BLOCK
00011-000012-PFAD:      PFLG
00012-000000 PFLG:      0               ;CONTINUE PROCESSING FLAG
00013-100000 ETMC:      100000          ;ELSYTEC MCA CODE 1000
00014-000377 RMSK:      377
00015-000010 TOM:       10              ;TIME OUT MASK
00016-000017 C15:       15.
00017-000000 CF:        0               ;RTC SET FOR EXTERNAL
00020-000240'.AMA:      AMEM
00021-000022-ZAD:       ZERO
00022-000000 ZERO:      0
00023-000000 CCNT1:     0
00024-000000 CCNT2:     0
00025-000127'ISRA:      ISR
00026-000000 AC0:       0
00027-000000 AC1:       0
00030-000000 AC2:       0
00031-000000 AC3:       0
00032-000104-BAT1A:     BAT1-1
00033-000144-BAT2A:     BAT2-1
00034-000104-IBTP:      BAT1-1
00035-000144-OBTP:      BAT2-1
00036-000006 SCF:       6
00037-000045-ISAD:      ISB
00040-000044-ISADD:     ISB-1
00041-000021-DAAD:      ZAD
00042-000000 DAS:       0
00043-000042-DASA:      DAS
00044-000000 AMRS:      0
        000040 ISB:     .BLK    32.
        000040 BAT1:    .BLK    32.
        000040 BAT2:    .BLK.   32.
                        .NREL
00000'060277 PAPB:      INTDS           ;DISABLE INTERRUPTS
00001'062677            IORST           ;RESET ALL I/O
00002'020017-           LDA     0.CF    ;SET CLOCK FREQ
00003'061014            DOA     0.RTC   ;TO EXTERNAL
00004'020003-           LDA     0.INTM  ;MASK OUT ALL BUT RTC
00005'062077            MSKO    0
00006'020025-           LDA     0.ISRA  ;SET UP INTERRUPT SERVICE ADDRESS
00007'040001            STA     0.1
```

```
0002  PAPB
 00010'102400           SUB     0,0
 00011'040002-          STA     0,XCFLG ;RESET XFER CYCLE FLAG
 00012'040004-          STA     0,SCNT  ;SET SAMPLE COUNT =0
 00013'020021-          LDA     0,ZAD   ;INITIALIZE D/A OUTPUT
 00014'040041-          STA     0,DAAD  ;ADDRESS FOR ZERO
 00015'020005-          LDA     0,PAAD  ;SET UP TO RECEIVE PARAMETERS
 00016'061007           DOA     0,MCAR  ;FROM MASTER COMPUTER
 00017'020006-          LDA     0,NP
 00020'100400           NEG     0,0
 00021'062107           DOBS    0,MCAR
 00022'063607           SKPDN   MCAR    ;WAIT TILL PARAMS RECEIVED
 00023'000777           JMP     .-1
 00024'060207           NIOC    MCAR    ;CLEAR DONE AND UNLOCK RECEIVER
 00025'006020-          JSR     @.AMA   ;ALLOCATE MEMORY
 00026'063077           HALT            ;MEMORY ALLOCATION ERROR
 00027'060477 CKSW:     READS   0       ;WAIT HERE TILL SW0 SET UP
 00030'101113           MOVL#   0,0,SNC
 00031'000776           JMP     CKSW
 00032'060114           NIOS    RTC     ;ENABLE CLOCK TO INTERRUPT
 00033'060177           INTEN
 00034'020002-CKXFG:    LDA     0,XCFLG ;IF XCFLG ALREADY SET,
 00035'101004           MOV     0,0,SZR ;LOOP TOOK TOO LONG, HALT
 00036'063077           HALT
 00037'020002-          LDA     0,XCFLG ;WAIT HERE TILL SXCFLG
 00040'101005           MOV     0,0,SNR ;SET DUE TO COMPLETE
 00041'000776           JMP     .-2     ;NEW BLOCK OF SAMPLES
 00042'126400           SUB     1,1
 00043'044002-          STA     1,XCFLG ;RESET XCFLG
 00044'060477           READS   0       ;IF SW0 UP, CONTINUE PROCESSING
 00045'101112           MOVL#   0,0,SZC ;AND SET PFLG=1
 00046'126000           ADC     1,1     ;ELSE, STOP PROCESSING
 00047'044012-          STA     1,PFLG  ;SET PFLG=0
 00050'020011-          LDA     0,PFAD  ;SEND PELG TO ELSYTEC
 00051'061006           DOA     0,MCAT
 00052'102000           ADC     0,0
 00053'062006           DOB     0,MCAT
 00054'020013-          LDA     0,ETMC
 00055'063106           DOCS    0,MCAT
 00056'063606           SKPDN   MCAT
 00057'000777           JMP     .-1
 00060'062406           DIC     0,MCAT  ;CHECK STATUS
 00061'030015-          LDA     2,TOM
 00062'113414           AND#    0,2,SZR ;DONE DUE TO TIME OUT?
 00063'063077           HALT            ;YES, LOOP IN A TOO LONG
 00064'125005           MOV     1,1,SNR ;IF PFLG=0,
 00065'000713           JMP     PAPB    ;STOP PROCESSING & WAIT FOR PARAMS
 00066'020007-          LDA     0,NCH   ;INITIALIZE CHANNEL
 00067'040023-          STA     0,CCNT1 ;COUNT TO NCH
 00070'104400           NEG     0,1
 00071'034037-          LDA     3,ISAD
 00072'070477           READS   2       ;GET # OF CHANNEL FOR D/A
 00073'151120           MOVZL   2,2     ;SKIP SW 0
 00074'151102           MOVL    2,2,SZC
 00075'000405           JMP     .+5
 00076'125404           INC     1,1,SZR
 00077'000775           JMP     .-3
 00100'034021-          LDA     3,ZAD   ;IF NO SWS SET UP,
 00101'000403           JMP     .+3     ;OUTPUT ZERO
 00102'123000           ADD     1,0     ;DISPLACEMENT IS SW#-1
```

```
0003  PAPB
 00103'117000           ADD     0.3      ;TO GET DESIRED CHANNEL
 00104'054041-          STA     3.DAAD   ;D/A OUTPUT ADDRESS
 00105'020035-          LDA     0.OBTP   ;INITIALIZE XFER ADDRESS POINTER
 00106'040020           STA     0.OADP   ;TO OUTPUT BUFFER ADD TABLE
 00107'020010-XFRLP:    LDA     0.NL     ;SET UP TO XMIT NL
 00110'100400           NEG     0.0      ;WORDS (2*NL PACKED SAMPLES)
 00111'062006           DOB     0.MCAT
 00112'022020           LDA     0.@OADP  ;GET NEXT XFER ADDRESS
 00113'061006           DOA     0.MCAT
 00114'020013-          LDA     0.ETMC   ;XMIT TO ELSYTEC MACHINE
 00115'063106 XMDB:     DOCS    0.MCAT   ;READ STATUS
 00116'063606           SKPDN   MCAT
 00117'000777           JMP     .-1
 00120'030015-          LDA     2.TOM
 00121'066606           DICC    1.MCAT
 00122'133414           AND#    1.2.SZR  ;DONE DUE TO TIME OUT?
 00123'000772           JMP     XMDB     ;YES. KEEP TRYING
 00124'014023-          DSZ     CCNT1    ;DONE NCH CHANNELS?
 00125'000762           JMP     XFRLP    ;NO. XMIT NEXT CHANNEL
 00126'000706           JMP     CKXFG    ;YES. WAIT FOR NEXT BLK
 00127'040026-ISR:      STA     0.AC0    ;SAVE ACS
 00130'044027-          STA     1.AC1
 00131'050030-          STA     2.AC2
 00132'054031-          STA     3.AC3
 00133'020037-          LDA     0.ISAD   ;SET UP A/D INTO ISB
 00134'061021           DOA     0.ADCV
 00135'020007-          LDA     0.NCH    ;SET UP A/D FOR NCH CHANS
 00136'040024-          STA     0.CCNT2
 00137'100400           NEG     0.0
 00140'062121           DOBS    0.ADCV
 00141'034041-          LDA     3.DAAD   ;ADDRESS OF SELECTED CHAN
 00142'102400           SUB     0.0
 00143'030036-          LDA     2.SCF
 00144'063621           SKPDN   ADCV     ;WAIT TILL A/D DONE
 00145'000777           JMP     .-1
 00146'025400           LDA     1.0.3    ;SCALE UP SELECTED SAMPLE
 00147'073301           MUL              ;FOR 12 BIT D/A
 00150'044042-          STA     1.DAS    ;STORE SAMPLE FOR D/A
 00151'020043-          LDA     0.DASA   ;SET D/A TO OUTPUT FROM
 00152'061023           DOA     0.DACV   ;DASA
 00153'102000           ADC     0.0
 00154'062123           DOBS    0.DACV   ;START OUTPUT TO D/A
 00155'063623           SKPDN   DACV
 00156'000777           JMP     .-1
 00157'060323           NIOP    DACV     ;CONVERT SAMPLE
 00160'020034-          LDA     0.IBTP   ;INITIALIZE INPUT ADDRESS
 00161'040021           STA     0.IADP   ;PTR TO INPUT BUFFER TABLE
 00162'020040-          LDA     0.ISADD  ;INITIALIZE INPUT SAMPLE PTR
 00163'040022           STA     0.ISDP   ;TO A/D BUFFER
 00164'030004-          LDA     2.SCNT
 00165'151222           MOVZR   2.2.SZC  ;SAMPLE COUNT ODD?
 00166'000413           JMP     RSLP     ;YES. PACK SAMPLES IN RIGHT HALF
 00167'024014-          LDA     1.RMSK   ;RIGHT BYTE 377 MASK
 00170'022022 LSLP:     LDA     0.@ISDP  ;GET NEXT CHAN SAMPLE
 00171'123700           ANDS    1.0      ;MASK OFF LOW 8 BITS
 00172'036021           LDA     3.@IADP  ;GET BUFFER BASE ADDRESS
 00173'157000           ADD     2.3      ;ADD SCNT/2 FOR DISPLACEMENT
 00174'041400           STA     0.0.3    ;STORE EVEN SAMPLE IN LEFT HALF
 00175'014024-          DSZ     CCNT2
```

```
0004  PAPB
 00176'000772           JMP     LSLP
 00177'151120           MOVZL   2,2       ;SHIFT IN 0 FOR EVEN SCNT
 00200'000414           JMP     ISC
 00201'024014-RSLP:     LDA     1,RMSK    ;RIGHT BYTE 377 MASK
 00202'022022           LDA     0,@ISDP   ;GET NEXT CHAN SAMPLE
 00203'123400           AND     1,0       ;MASK OFF LOW 8 BITS
 00204'036021           LDA     3,@IADP   ;GET BUFFER BASE ADDRESS
 00205'157000           ADD     2,3       ;ADD SCNT/2 FOR DISPLACENE
 00206'025400           LDA     1,0,3     ;GET PREVIOUS SAMPLE
 00207'123000           ADD     1,0       ;PACK ODD SAMPLE IN RIGHT HALF
 00210'041400           STA     0,0,3
 00211'014024-          DSZ     CCNT2
   00212'000767           JMP     RSLP
 00213'151140           MOVOL   2,2       ;SHIFT IN 1 FOR ODD SCNT
 00214'151400 ISC:      INC     2,2       ;INCREMENT SAMPLE COUNT
 00215'020010-          LDA     0,NL      ;# FREQ LINES
 00216'101120           MOVZL   0,0       ;2*NL=# SAMPLES/BLOCK
 00217'142404           SUB     2,0,SZR   ;IS SCNT=2*NL?
 00220'000410           JMP     SSCT      ;NO. SAVE SCNT & RESTORE STATUS
 00221'152400           SUB     2,2       ;YES. SET SCNT=0
 00222'102000           ADC     0,0       ;SET XFER CYCLE FLAG
 00223'040002-          STA     0,XCFLG
 00224'020035-          LDA     0,OBTP    ;SWAP OUTPUT BUFFER
 00225'024034-          LDA     1,IBTP    ;AND INPUT BUFFER TABLE PTRS
 00226'044035-          STA     1,OBTP    ;TO INPUT TO PREVIOUS PROC BFR
 00227'040034-          STA     0,IBTP    ;AND OUT NEW SAMPLE BLOCK
 00230'050004-SSCT:     STA     2,SCNT    ;SAVEJMAMPLE COUNT
 00231'020026-          LDA     0,AC0     ;RESTORE ACS
 00232'024027-          LDA     1,AC1
 00233'030030-          LDA     2,AC2
 00234'034031-          LDA     3,AC3
 00235'060114           NIOS    RTC
 00236'060177           INTEN             ;REENABLE INTERRUPT
 00237'002000           JMP     @0        ;RETURN TO MAIN PROG
 00240'054044-AMEM:     STA     3,AMRS
 00241'020032-          LDA     0,BAT1A   ;INIT USE BAT1 BUFFER ADD
 00242'040034-          STA     0,IBTP    ;TABLE FOR INPUT
 00243'040021           STA     0,IADP
 00244'020033-          LDA     0,BAT2A   ;USE BAT2 BUFFER ADD
 00245'040035-          STA     0,OBTP    ;TABLE FOR OUTPUT
 00246'040020           STA     0,OADP
 00247'020007-          LDA     0,NCH
 00250'100400           NEG     0,0
 00251'024001-          LDA     1,MUPL    ;MEMORY UPPER LIMIT
 00252'030010-          LDA     2,NL      ;# LINES = BUFFER SIZE
 00253'034000-          LDA     3,BLL     ;BUFFER LOWER LIMIT
 00254'056021 IBAL:     STA     3,@IADP   ;STORE NEXT INPUT BUFFER
 00255'157000           ADD     2,3       ;ADDRESS IN TABLE
 00256'166512           SUBL#   3,1,SZC   ;IS MEMORY ALLOCATED >MUL?
 00257'002044-          JMP     @AMRS     ;YES. CAN'T ALLOCATE ENOUGH MEMORY
 00260'101404           INC     0,0,SZR
 00261'000773           JMP     IBAL
 00262'020007-          LDA     0,NCH
 00263'100400           NEG     0,0
 00264'056020 OBAL:     STA     3,@OADP   ;STORE NEXT OUTPUT BUFFER
 00265'157000           ADD     2,3       ;ADDRESS IN TABLE
 00266'166512           SUBL#   3,1,SZC   ;IS MEMORY ALLOCATED >MUL?
 00267'002044-          JMP     @AMRS     ;YES. CAN'T ALLOCATE ENOUGH MEMORY
 00270'101404           INC     0,0,SZR
```

```
0005  PAPB
 00271'000773          JMP     OBAL
 00272'010044-         ISZ     AMRS      ;ALLOCATION SUCCESSFUL
 00273'002044-         JMP     @AMRS     ;RETURN TO JSR+2
       000000'         .END    PAPB
```

```
---
0001  PAPA

                ; 9/4/75          C. SHIVELY
                ;                ARRAY PROCESSING
                ; PROGRAM FOR SPECTRUM ANALYSIS
                ; IN ELSYTEC MACHINE A
                ; SAME AS 8/20/74. EXCEPT DISPLAY
                ; CORRECTED FOR NL DIFFERENT FROM 512
                ; DISPLAYS FIRST 1024 LINES IF NL>512
                ; AND ONLY NL IF NL <512
                ; REQUIRES ELSYTEC SUBROUTINES
                ; HFFTI. CALLS. MATHS. SCPLD
                ; SENDS DATA IN BLOCK FLOATING POINT
                ; NLR COMPLEX LINES FOLLOWED BY EXPONENT
                ; BASE 2 AND 1+EXP OF LARGEST MANTISSA.
                ; I.E. -# LEFT SHIFTS NEEDED TO NORMALIZE
                ; LARGEST MANTISSA.
                ; DATA ARE 15 BIT FRACTIONS WITH BINARY POINT
                ; BETWEEN SIGN BIT 0 AND BIT 1
                ; DATA WILL BE NORMALIZED SO THAT MAGNITUDE
                ; OF LARGEST NUMBER IS BETWEEN 1/2 AND 1.
                ; AND FLAG FOLLOWING EXP IS THEREFORE 0.
                ; UNLESS DATA IS ALL 0. IN WHICH CASE NORMALIZATION
                ; HAS FAILED. AND FLAG = -4
                        .TITL   PAPA
                        .ENT    CTABP.C1SAV.C2SAV.TDFLG.MXLN.MKFLG
                        .ENT    AVFLG.ALLNQ.MRKSQ.TRANQ.PLOTQ.AUTO.
                        .ENT    .OBUF..NLN..SCL..FLG.PLFLG
                        .ENT    .CHAN..ALAD..SNPD..OBEB..OZRO..XLGX
                        .ENT    .SCLW..INTF..DWD..MLN..NOSH
                        .ENT    AOFLG.SEFLG
                        .EXTD   LDRAM.LNUM.CARP.FFTC.SCPLD.ARGOT.CNTF
      000000            .NLN =  0
      000001            .OBUF = 1
      000002            .FLG =  2
      000003            .SCL =  3
      000004            .CHAN = 4
      000005            .ALAD = 5
      000006            .SNPD = 6
      000007            .OBEB = 7
      000010            .OZRO = 10
      000011            .XLGX = 11
      000012            .SCLW = 12
      000013            .INTF = 13
      000014            .DWD =  14
      000015            .MLN =  15
      000016            .NOSH = 16
      000030            .LOC    30
00030 000000 UPIP:      0
00031 000000 UPOP:      0
                        .ZREL
00000-003477 OBAD:      3477            ;OUTPUT BUFFER ADDRESS
                ; MUST BE GREATER THAN NMAX
00001-000003-PAAD:      NCH
00002-000004 NP:        4
00003-000013 NCH:       11.
00004-002000 NL:        1024.
00005-000144 NLR:       100.
00006-000000 RLD:       0
00007-002000 NL1:       1024.
```

```
0002  PAPA
 00010-007577 WLLM:    7577
 00011-000000 FLGM:    0
 00012-000000 SCLM:    0
 00013-000014-PFAD:    PFLG
 00014-000000 PFLG:    0
 00015-000000 MTCRF:   0
 00016-020000 TMMC:    20000          ;MAG TAPE MACHINE MCA CODE
 00017-000010 TOM:     10
 00020-000000 CCNT:    0
 00021-000000 CFD:     0
 00022-000007-CTABP:   NL1
 00023-000007-NLAD:    NL1
 00024-000304'UNPK:    UNPAK
 00025-000000 UPRTN:   0
 00026-000000 UPLCT:   0
 00027-000000 PLFLG:   0
 00030-000377 RMSK:    377
 00031-000000 SEFLG:   0
 00032-000000 AOFLG:   0
 00033-177777 TDFLG:   -1
 00034-000011 MXLN:    11
 00035-000000 MKFLG:   0
 00036-000275'AUTO.:   AUTOQ
 00037-000000 AVFLG:   0
 00040-000042-C1SAV:   C1TAB
 00041-000061-C2SAV:   C2TAB
 00042-001000 C1TAB:   512.
 00043-017777 OBUF1:   17777
 00044-000000 FLG1:    0
 00045-000000 SCL1:    0
 00046-000001 CHAN1:   1
 00047-000000 ALAD1:   0
 00050-177777 SNPD1:   -1
 00051-000000 OBEB1:   0
 00052-000000 OZRO1:   0
 00053-177772 XLGX1:   177772
 00054-000000 SCLW1:   0
 00055-000010 INTF1:   10
 00056-000403 DWD1:    403            ;LIN. 24DB=120403.48DB=140403
 00057-001000 MLN1:    512.
 00060-000000 NOSH1:   0
 00061-001000 C2TAB:   512.
 00062-021777 OBUF2:   21777
 00063-000000 FLG2:    0
 00064-000000 SCL2:    0
 00065-000002 CHAN2:   2
 00066-000000 ALAD2:   0
 00067-177777 SNPD2:   -1
 00070-000000 OBEB2:   0
 00071-000000 OZRO2:   0
 00072-177772 XLGX2:   177772
 00073-000000 SCLW2:   0
 00074-000010 INTF2:   10
 00075-000403 DWD2:    403
 00076-001000 MLN2:    512.
 00077-000000 NOSH2:   0
 00100-000403 DWDLN:   403           .:LINEAR. ALL LINES
 00101-140403 DWDLG:   140403         :LOG 48 DB. 24DB=120403.96DB=160403
 00102-001000 CS12:    512.
```

```
00003  PAPA
 00103-000061-C2TBA:   C2TAB
 00104-177772 XLX:     177772
 00105-177774 MFOUR:   -4
 00106-000000 SCLF:    0
 00107-000000 FLGF:    0
 00110-000000 TEM:     0
 00111-000000 TNLR:    0
 00112-000047'RCVP:    RCVPF
                       .NREL
 00000'060277 PAPA:    INTDS             ;DISABLE INTERRUPTS
 00001'062677          IORST             ;RESET I/O
 00002'006001$         JSR       @LDRAM  ;LOAD FFT RAM PROG
 00003'020001-         LDA       0.PAAD  ;SET UP TO RECEIVE PARAMETERS
 00004'061007          DOA       0.MCAR  ;FROM MASTER COMPUTER
 00005'020002-         LDA       0.NP
 00006'100400          NEG       0.0
 00007'062107          DOBS      0.MCAR
 00010'063607          SKPDN     MCAR
 00011'000777          JMP       .-1     ;WAIT TILL PARAMS RECEIVED
 00012'060207          NIOC      MCAR    ;UNLOCK RECEIVER & CLEAR DONE
 00013'102400          SUB       0.0     ;RESET SETUP AND ARITHMETIC
 00014'040032-         STA       0.AOFLG ;ERROR FLAGS
 00015'040031-         STA       0.SEFLG
 00016'024010-         LDA       1.WLLM  ;SET CHAN 1 DISP ADDRESS TO
 00017'044043-         STA       1.OBUF1 ;WLLM
 00020'030102-         LDA       2.C512  ;SET CHAN 2 DISP ADDRESS
 00021'147000          ADD       2.1     ;TO WLLM+1024 TO DISP
 00022'147000          ADD       2.1     ;SECOND 512 LINES IF
 00023'044062-         STA       1.OBUF2 ;MORE THAN 512 LINES
 00024'024004-         LDA       1.NL    ;COPY # LINES TO NL1
 00025'044007-         STA       1.NL1
 00026'132513          SUBL#     1.2.SNC ;IS NL>512
 00027'000403          JMP       .+3
 00030'145000          MOV       2.1     ;YES. USE 512 AND
 00031'020103-         LDA       0.C2TBA ;OUTPUT NEXT 512 LINES ON CH2
 00032'040041-         STA       0.C2SAV
 00033'044042-         STA       1.C1TAB ;SET UP # OF LINES FOR OUTPUT
 00034'044061-         STA       1.C2TAB ;ON SCOPE
 00035'044057-         STA       1.MLN1
 00036'044076-         STA       1.MLN2
 00037'020104-         LDA       0.XLX   ;177772 XLX WORD FOR NL=512
 00040'132415          SUB#      1.2.SNR ;FORM XLX WD BITS 13-15 FOR NL
 00041'000404          JMP       .+4     ;2 FOR 512. 3 FOR 256. 4 FOR 128.
 00042'125120          MOVZL     1.1     ;5 FOR 64. 6 FOR 32
 00043'101400          INC       0.0
 00044'000774          JMP       .-4
 00045'040053-         STA       0.XLGX1
 00046'040072-         STA       0.XLGX2
 00047'020013-RCVPF:   LDA       0.PFAD  ;SET UP TO RECEIVE
 00050'061007          DOA       0.MCAR  ;PROCESS FLAG FROM A/D
 00051'126000          ADC       1.1     ;MACHINE B
 00052'066107          DOBS      1.MCAR
 00053'063607          SKPDN     MCAR    ;WAIT TILL PFLG RECEIVED
 00054'000777          JMP       .-1
 00055'061006          DOA       0.MCAT  ;SET UP TO SEND PFLG
 00056'066006          DOB       1.MCAT  ;TO TAPE MACHINE C
 00057'030016-         LDA       2.TMMC
 00060'073106          DOCS      2.MCAT
 00061'063606          SKPDN     MCAT
```

```
0004  PAPA
  00062'000777          JMP     .-1
  00063'062606          DICC    0.MCAT  ;CHECK STATUS
  00064'030017-         LDA     2.TOM
  00065'126000          ADC     1.1
  00066'113414          AND#    0.2.SZR ;DONE DUE TO TIME OUT?
  00067'126400          SUB     1.1     ;YES. SET MTCRF=0 TO SKIP
  00070'044015-         STA     1.MTCRF ;XMIT TO MAG TAPE MACHINE
  00071'020014-         LDA     0.PFLG
  00072'101005          MOV     0.0.SNR ;IF PFLG=0.
  00073'000705          JMP     PAPA    ;STOP PROC & WAIT FOR PARAMS
  00074'020003-         LDA     0.NCH   ;SET CNT TO # OF CHANS
  00075'040020-         STA     0.CCNT
  00076'104400          NEG     0.1
  00077'070477          READS   2       ;GET # OF CHANNEL FOR D/A
  00100'151120          MOVZL   2.2     ;SKIP SW0
  00101'151102          MOVL    2.2.SZC
  00102'000404          JMP     .+4
  00103'125404          INC     1.1.SZR
  00104'000775          JMP     .-3
  00105'105001          MOV     0.1.SKP ;IF NO SWS SET. DISPLAY CHAN 1
  00106'124400          NEG     1.1
  00107'044021-         STA     1.CFD   ;NCH+1-CH# FOR DISPLAY
  00110'020023-         LDA     0.NLAD  ;SET CTABP TO MAIN
  00111'040022-         STA     0.CTABP ;WORK AREA PARAMETERS
  00112'020010-RCVLP:   LDA     0.WLLM  ;WORK AREA LOWER LIMIT
  00113'101400          INC     0.0     ;=BUFFER ADDRESS -1
  00114'061007          DOA     0.MCAR
  00115'020004-         LDA     0.NL
  00116'100400          NEG     0.0
  00117'062107          DOBS    0.MCAR
  00120'063607          SKPDN   MCAR    ;WAIT TILL NEXT DATA
  00121'000777          JMP     .-1     ;BLOCK RECEIVED
  00122'006024-         JSR     @UNPK   ;UNPACK SAMPLES INTO LEFT HALF
  00123'030010-         LDA     2.WLLM
  00124'024004-         LDA     1.NL    ;# TIME SAMPLES=2*NL
  00125'125120          MOVZL   1.1
  00126'102400          SUB     0.0     ;ADJACENT LOCS
  00127'006002$         JSR     @LNUM   ;FIND LARGEST NUMBER IN ORDER
  00130'044011-         STA     1.FLGM  ;TO SET FFT INIT SCALE FLAG
  00131'152400          SUB     2.2     ;TAKE FORWARD XFRM
  00132'102520          SUBZL   0.0     ;SET INITIAL
  00133'040012-         STA     0.SCLM  ;SCALE =1
  00134'006004$         JSR     @FFTC   ;OF NEXT DATA BLOCK
  00135'020015-         LDA     0.MTCRF
  00136'101005          MOV     0.0.SNR ;IS MAG TAPE COMPUTER RESPONDING?
  00137'000505          JMP     CDC     ;NO. DON'T SEND DATA TO IT
  00140'024005-         LDA     1.NLR   ;SEARCH 2*NLR ADJACENT
  00141'125120          MOVZL   1.1     ;LOCS IN WORK AREA
  00142'044111-         STA     1.TNLR  ;FOR LARGEST MAGNITUDE
  00143'030006-         LDA     2.RLD   ;TO BE RETURNED
  00144'151120          MOVZL   2.2
  00145'020010-         LDA     0.WLLM
  00146'113000          ADD     0.2
  00147'050110-         STA     2.TEM   ;SAVE ADDR OF BLOCK TO BE OUTPUT
  00150'102400          SUB     0.0
  00151'006002$         JSR     @LNUM   ;2**FLGF>LM>=2**(FLGF-1)
  00152'044107-         STA     1.FLGF  ;I.E. FLGF = # OF LEFT SHIFTS TO NOMALIZE
  00153'020012-         LDA     0.SCLM  ;TRUE=CORE*2**(-SCLM)
  00154'100400          NEG     0.0     ;TRUE=CORE*2**(EXP)
```

```
0005  PAPA
 00155'123000          ADD     1.0       ;CORRECT EXP FOR NORMALIZATION SHIFTS
 00156'040106-         STA     0.SCLF
 00157'020111-         LDA     0.TNLR    ;MOVE NLR LINES TO OUTPUT
 00160'024110-         LDA     1.TEM     ;BUFFER SHIFTING LEFT /FLGF/PLACES
 00161'030000-         LDA     2.OBAD    ;TO NORMALIZE DATA
 00162'006007$         JSR     @CNTF
 00163'000107-         FLGF
 00164'020105-         LDA     0.MFOUR   ;SET COUNTER TO NORMALIZE
 00165'040110-         STA     0.TEM     ;ONLY FOUR TIMES
 00166'020105-CN:      LDA     0.MFOUR   ;IF FLGF WAS NOT -4.
 00167'024107-         LDA     1.FLGF    ;THEN NORM DONE
 00170'122405          SUB     1.0.SNR
 00171'000403          JMP     .+3
 00172'126400          SUB     1.1       ;SET FLGF=0 NORM DONE
 00173'000423          JMP     MEF-1
 00174'030000-         LDA     2.OBAD    ;SEARCH OUTPUT BUFFER
 00175'024111-         LDA     1.TNLR    ;FOR LARGEST AGAIN
 00176'102400          SUB     0.0
 00177'006002$         JSR     @LNUM
 00200'125005          MOV     1.1.SNR   ;IF FLGF NOW 0.
 00201'000415          JMP     MEF-1     ;NORM DONE
 00202'010110-         ISZ     TEM       ;IF ALREADY SCALED UP
 00203'000402          JMP     .+2       ;4 TIMES. QUIT
 00204'000413          JMP     MEF
 00205'020106-         LDA     0.SCLF    ;CORRECT EXP BY SUBTRACTING
 00206'123000          ADD     1.0       ;# LEFT SHIFTS TO BE DONE
 00207'040106-         STA     0.SCLF    ;TO NOMALIZE
 00210'020111-         LDA     0.TNLR
 00211'024000-         LDA     1.OBAD
 00212'131000          MOV     1.2
 00213'006007$         JSR     @CNTF     ;SHIFT LEFT IN OUTPUT TO NORMALIZE
 00214'000107-         FLGF
 00215'000751          JMP     CN
 00216'044107-         STA     1.FLGF
 00217'030000-MEF:     LDA     2.OBAD
 00220'151400          INC     2.2
 00221'071006          DOA     2.MCAT    ;SET UP TO SEND FROM OUTPUT BUFFER
 00222'020111-         LDA     0.TNLR
 00223'113000          ADD     0.2       ;ADDR OF EXP
 00224'101400          INC     0.0
 00225'101400          INC     0.0
 00226'100400          NEG     0.0       ;SEND 2*NLR+2 WORDS
 00227'062006          DOB     0.MCAT
 00230'020106-         LDA     0.SCLF    ;SEND EXP AS 1ST WORD AFTER DATA
 00231'041000          STA     0.0.2
 00232'020107-         LDA     0.FLGF    ;SEND FLAG AS 2ND WD AFTER DATA
 00233'041001          STA     0.1.2
 00234'024016-         LDA     1.TMMC
 00235'067106 XMDC:    DOCS    1.MCAT    ;XMIT TO MAG TAPE MACH
 00236'063606          SKPDN   MCAT
 00237'000777          JMP     .-1
 00240'062606          DICC    0.MCAT    ;READ STATUS
 00241'030017-         LDA     2.TOM
 00242'113414          AND#    0.2.SZR   ;DONE DUE TO TIME OUT?
 00243'000772          JMP     XMDC      ;YES. KEEP TRYING
 00244'020020-CDC:     LDA     0.CCNT
 00245'024021-         LDA     1.CFD
 00246'122404          SUB     1.0.SZR   ;IS THIS CHAN ONE FOR DISPLAY?
 00247'000423          JMP     DCT       ;NO
```

```
0006  PAPA
 00250'102400           SUB     0.0     ;YES. CLEAR POLAR
 00251'040027-          STA     0.PLFLG ;COORDINATE FLAG
 00252'006003$          JSR     @CARP   ;AND CONVERT TO POLAR
 00253'020012-          LDA     0.SCLM  ;MOVE SCALE TO DISPLAY
 00254'040060-          STA     0.NOSH1 ;PARAM TABLES FOR #
 00255'040077-          STA     0.NOSH2 ;OF SHIFTS ON OUTPUT
 00256'060477           READS   0       ;IF SW0 IS DOWN.
 00257'024100-          LDA     1.DWDLN ;SET UP FOR LINEAR DISPLAY
 00260'101132           MOVZL#  0.0.SZC ;IF SW0 IS UP.
 00261'024101-          LDA     1.DWDLG ;SET UP FOR 48 DB LOG DISPLAY
 00262'044056-          STA     1.DWD1
 00263'044075-          STA     1.DWD2
 00264'006005$          JSR     @SCPLD  ;LOAD DISPLAY BUFFERS AND FLAGS
 00265'102400           SUB     0.0     ;RESET SETUP AND
 00266'040032-          STA     0.AOFLG ;ARITHMETIC ERROR FLAGS
 00267'040031-          STA     0.SEFLG
 00270'020023-          LDA     0.NLAD  ;RESTORE CTABP
 00271'040022-          STA     0.CTABP ;TO MAIN WORK AREA
 00272'014020-DCT:      DSZ     CCNT    ;PROCESS NEXT CHANNEL
 00273'000617           JMP     RCVLP
 00274'002112-          JMP     @RCVP   ;WAIT FOR PROCESS FLAG
 00275'001400 AUTOQ:    JMP     0.3     ;DUMMY NO AUTO SCALE
 00276'001401 ALLNQ:    JMP     1.3     ;DUMMY ALL LINES
 00277'102400 MRKSQ:    SUB     0.0     ;DUMMY NO MARKS
 00300'040035-          STA     0.MKFLG ;RESET MKFLG
 00301'001400           JMP     0.3
 00302'001400 TRANQ:    JMP     0.3     ;DUMMY NO TRANSIENT
 00303'001400 PLOTQ:    JMP     0.3     ;DUMMY NO PLOT
 00304'054025-UNPAK:    STA     3.UPRTN ;SAVE RETURN ADDRESS
 00305'020004-          LDA     0.NL
 00306'040026-          STA     0.UPLCT
 00307'105120           MOVZL   0.1
 00310'030010-          LDA     2.WLLM  ;BUFFER ADDRESS-1
 00311'151400           INC     2.2
 00312'143000           ADD     2.0     ;SET INPUT AUTO DEC
 00313'040030           STA     0.UPIP  ;TO BAD=+NL
 00314'147000           ADD     2.1     ;SET OUTPUT AUTO DEC
 00315'044031           STA     1.UPOP  ;TO BAD+2*NL
 00316'024030-          LDA     1.RMSK  ;377 RIGHT BYTE MASK
 00317'131000 UPLP:     MOV     1.2     ;RIGHT BYTE MASK IN AC2
 00320'135300           MOVS    1.3     ;LEFT BYTE MASK IN AC3
 00321'022030           LDA     0.@UPIP ;GET PCKED SMPLES IN REVERSE
 00322'113700           ANDS    0.2     ;MASK RIGHT BYTE INTO LEFT
 00323'052031           STA     2.@UPOP ;AT HIGHER MEM LOC
 00324'117400           AND     0.3     ;MASK LEFT IN LEFT
 00325'056031           STA     3.@UPOP ;STORE IN LOWER MEM LOC
 00326'014026-          DSZ     UPLCT
 00327'000770           JMP     UPLP
 00330'002025-          JMP     @UPRTN
      000000'           .END    PAPA
```

```
---
0001  CNTF

                ; 10/16/73      C. SHIVELY
                ; SUBROUTINE TO SCALE SIGNED
                ; NUMBERS BY POWERS OF 2
                ; USING RIGHT SHIFTS (NRS>0)
                ; OR LEFT SHIFTS (NRS<0)
                ; IF NRS=0. MERELY MOVES DATA
                ; CALLING SEQUENCE:
                ;       AC0 CONTAINS # OF WORDS
                ;       AC1 CONTAINS STARTING ADDR-1
                ;       AC2 CONTAINS OUTPUT ADDR-1
                ;       JSR     @CNTF
                ;       NRSA            ;ADDRESS OF NRS
                ;       (RETURN)
                ; SETS BIT 9 IN AOFLG=1 IF OVERFLOW
                ; OR BIT 9=0 IF NO OVERFLOW
                ; SATURATES MAGNITUDES AT 77777 OCTAL
                        .TITL   CNTF
        000020          OPT = 20
                        .ENT    CNTF.CNTFA
                        .EXTD   AOFLG
                        .ZREL
00000-000000'CNTF:      CNTFA
                        .NREL
00000'050020 CNTFA:     STA     2.OPT   ;SET OUTPUT AUTO INC
00001'131000            MOV     1.2     ;INPUT ADDR TO AC2
00002'040473            STA     0.LCNT
00003'023400            LDA     0.@0.3  ;GET # RIGHT SHIFTS
00004'175400            INC     3.3
00005'054466            STA     3.RTNC  ;SAVE RETURN ADDRESS
00006'101004            MOV     0.0.SZR ;IS NRS=0?
00007'000407            JMP     CNS     ;NO. DO SHIFTS
00010'021001 MVLP:      LDA     0.1.2   ;YES. JUST MOVE DATA
00011'042020            STA     0.@OPT
00012'151400            INC     2.2
00013'014462            DSZ     LCNT
00014'000774            JMP     MVLP
00015'002456            JMP     @RTNC
00016'101112 CNS:       MOVL#   0.0.SZC ;LEFT SHIFTS?
00017'000422            JMP     LSLP-1  ;YES
00020'100400            NEG     0.0     ;NO. SHIFT RIGHT
00021'040453            STA     0.SCNT
00022'034452 RSLP:      LDA     3.SCNT
00023'021001            LDA     0.1.2   ;NEXT SIGNED NUMBER
00024'101113            MOVL#   0.0.SNC
00025'105001            MOV     0.1.SKP
00026'104400            NEG     0.1     ;MAGNITUDE IN AC1
00027'125220            MOVZR   1.1     ;SHIFT RIGHT
00030'175404            INC     3.3.SZR
00031'000776            JMP     .-2
00032'101112            MOVL#   0.0.SZC ;CORRECT SIGN
00033'124400            NEG     1.1
00034'046020            STA     1.@OPT
00035'151400            INC     2.2
00036'014437            DSZ     LCNT
00037'000763            JMP     RSLP
00040'002433            JMP     @RTNC
00041'040433            STA     0.SCNT
00042'034432 LSLP:      LDA     3.SCNT
```

```
0002  CNTF
 00043'021001           LDA     0,1,2   ;NEXT SIGNED NUMBER
 00044'101113           MOVL#   0,0,SNC
 00045'105001           MOV     0,1,SKP
 00046'104400           NEG     0,1     ;MAGNITUDE IN AC1
 00047'125122           MOVZL   1,1,SZC ;SHIFT LEFT
 00050'000414           JMP     OVFL    ;OVERFLOW OUT OF SIGN
 00051'175404           INC     3,3,SZR
 00052'000775           JMP     .-3
 00053'125112           MOVL#   1,1,SZC ;OVERFLOW INTO SIGN?
 00054'000410           JMP     OVFL    ;YES
 00055'101112 CSL:      MOVL#   0,0,SZC ;CORRECT SIGN
 00056'124400           NEG     1,1
 00057'046020           STA     1,@OPT
 00060'151400           INC     2,2
 00061'014414           DSZ     LCNT
 00062'000760           JMP     LSLP
 00063'002410           JMP     @RTNC
 00064'034412 OVFL:     LDA     3,OMSK  ;177677
 00065'024001$          LDA     1,AOFLG
 00066'167400           AND     3,1     ;TURN ON BIT 9
 00067'166000           ADC     3,1     ;IN OVERFLOW FLAG
 00070'044001$          STA     1,AOFLG
 00071'126220           ADCZR   1,1     ;SATURATE AT MAX =77777
 00072'000763           JMP     CSL
 00073'000000 RTNC:     0
 00074'000000 SCNT:     0
 00075'000000 LCNT:     0
 00076'177677 OMSK:     177677
                        .END
```

```
0001  RMDFE

                ; 9/4/75          C. SHIVELY
                ; REVISED TO RECEIVE BLOCK FLOATING POINT
                ; FROM PAPABFP.  BE SURE BASIC ALLOCATES N+1
                ; ELEMENTS FOR N ELEMENT ARRAY
                ; BASIC SUBROUTINES TO RECEIVE DIFAR OR
                ; OTHER MULTI-CHANNEL DATA FROM ELSYTEC
                ; VIA MCA FROM              ARRAY PROCESSING
                ; PROGRAM PAPA
                        .ENT    IRMD
                ;       CALL 11.C.B.L.FØ
                ; C IS # OF CHANNELS
                ; B IS # OF LINES IN TOTAL SPECTRUM
                ; L IS # OF LINES RETURNED TO BASIC
                ; FØ IS CELL # OF FIRST LINE RETURNED
                ; THIS ENTRY POINT INITIALIZES THE
                ; PARAMETERS NCH.NL.LR. & RLD IN
                ; PAPA (ELSYTEC PROG) AND IN PAPB
                ; (A/D MACHINE B PROG).  BEFORE CALLING
                ; 11. BOTH THESE PROGS MUST BE IN
                ; RECEPTIVE STATE. I.E. FOLLOWING INITIAL
                ; LOADING. OR AFTER TERMINATION OF A RUN
                ; BY SETTING SWØ DOWN ON CPU B. OR BY
                ; RESTART OF BOTH PROGS IN LOC 1000 (8).
                        .ENT    RMDFE
                ;       CALL 12.R(Ø).I(Ø).P
                ; FOLLOWING INITIALIZATION.SWØ ON CPUB MAY BE
                ; SET UP TO BEGIN PROCESSING IN CPU'S
                ; A & B. SUBSEQUENT CALLS TO 12 WILL RETURN
                ; COMPLEX SPECTRAL DATA REAL PARTS IN ARRAY
                ; R AND IMAG PARTS IN ARRAY I IN ORDER OF
                ; PROCESSING. CHAN 1. CHAN 2. CHAN3. ETC.
                ; RETURNS PROCESSING FLAG IN P
                ; IF P=Ø. RUN TERMINATED BY SWØ DOWN ON CPU B
                ; AND INITIALIZATION POSSIBLE
                ; IF P=-1. PROCESSING IS CONTINUING
                        .EXTD   .FLOT..FIX
                        .EXTN   BFLTM
                ; USES BFLT TO CONVERT FROM BLOCK FLOATING POINT
                ; TO DATA GENERAL FLOATING POINT
                        .TITL   RMDFE
                        .NREL
00000'054445 IRMD:      STA     3.RTN
00001'004435            JSR     CTI     ;CONVERT # CHANS TO FIXED
00002'044452            STA     1.NCH
00003'004433            JSR     CTI     ;CONVERT # LINES TO FIXED
00004'044451            STA     1.NL
00005'004431            JSR     CTI     ;CONVERT # LINES RETURNED TO FIXED
00006'044450            STA     1.NLR
00007'004427            JSR     CTI     ;CONVERT 1ST LINE CELL TO FIXED
00010'044447            STA     1.RLD
00011'030435            LDA     2.PAAD  ;SEND PARAMS
00012'024437            LDA     1.MNPFA ;NCH.NL.NLR.RLD
00013'020437            LDA     Ø.ETMC  ;TO ELSYTEC
00014'004406            JSR     XPAM
00015'030431            LDA     2.PAAD  ;SEND PARAMS
00016'024432            LDA     1.MNPFB ;NCH.NL
00017'020434            LDA     Ø.ADMC  ;TO A/D MACHINE B
00020'004402            JSR     XPAM
```

```
0002  RMDFE
 00021'002424           JMP     @RTN
 00022'071006 XPAM:     DOA     2.MCAT  ;ADDRESS
 00023'066006           DOB     1.MCAT  ;WORD COUNT
 00024'030423           LDA     2.TOM
 00025'060277 XMP:      INTDS           ;DISABLE INTERRUPTS
 00026'063106           DOCS    0.MCAT  ;TRY TO XMIT
 00027'063606           SKPDN   MCAT
 00030'000777           JMP     .-1
 00031'066606           DICC    1.MCAT  ;READ STATUS & CLEAR DONE
 00032'060177           INTEN           ;ALLOW BASIC TO INTERRUPT
 00033'133414           AND#    1.2.SZR ;TIME OUT?
 00034'000771           JMP     XMP     ;YES. TRY AGAIN
 00035'001400           JMP     0.3
 00036'054422 CTI:      STA     3.S3
 00037'032406           LDA     2.@RTN  ;GET NEXT PARAM ADDRESS
 00040'021000           LDA     0.0.2
 00041'025001           LDA     1.1.2
 00042'006002$          JSR     @.FIX   ;CONVERT TO FIXED POINT
 00043'010402           ISZ     RTN
 00044'002414           JMP     @S3
 00045'000000 RTN:      0
 00046'000054'PAAD:     NCH
 00047'000010 TOM:      10
 00050'177776 MNPFB:    -2
 00051'177774 MNPFA:    -4
 00052'100000 ETMC:     100000
 00053'040000 ADMC:     40000
 00054'000003 NCH:      3
 00055'002000 NL:       1024.
 00056'000100 NLR:      64.
 00057'000000 RLD:      0
 00060'000000 S3:       0
 00061'054537 RMDFE:    STA .   3.RTN1
 00062'021400           LDA     0.0.3   ;ADDR OF REAL ARRAY
 00063'040530           STA     0.RA
 00064'021401           LDA     0.1.3   ;ADDR OF IMAG ARRAY
 00065'040527           STA     0.IA
 00066'021402           LDA     0.2.3   ;ADDR OF P FLAG
 00067'040526           STA     0.PAD
 00070'060277           INTDS           ;DISABLE INTERRUPTS
 00071'020525           LDA     0.PFAD  ;SET UP MCA TO RECEIVE
 00072'061007           DOA     0.MCAR  ;FLAG FROM ELSYTEC
 00073'102000           ADC     0.0
 00074'062107           DOBS    0.MCAR
 00075'063607           SKPDN   MCAR
 00076'000777           JMP     .-1
 00077'060207           NIOC    MCAR    ;CLEAR DONE
 00100'060177           INTEN           ;ALLOW INTERRUPT
 00101'020516           LDA     0.PFLG  ;IF PROCESS FLAG =0.
 00102'101005           MOV     0.0.SNR ;RETURN WITH NO DATA
 00103'000475           JMP     BACK
 00104'020750           LDA     0.NCH   ;SET LCNT TO # CHANS
 00105'040505           STA     0.LCNT
 00106'020505           LDA     0.RA    ;START RECEIVING SPECTRAL LINES
 00107'040512           STA     0.INPAD ;AT BEGINNING OF REAL ARRAY
 00110'020504           LDA     0.IA    ;START STORING IMAG PARTS
 00111'040511           STA     0.CIAD  ;AT BEGINNING OF IMAG ARRAY
 00112'020507 RCVLP:    LDA     0.INPAD ;SET UP MCA TO RECEIVE NEXT CHAN
 00113'061007           DOA     0.MCAR
```

```
0003  RMDFE
 00114'024742           LDA     1.NLR
 00115'125400           INC     1.1     ;RECEIVE NLR LINES. EXP AND FLAG TOO
 00116'124520           NEGZL   1.1
 00117'060277           INTDS           ;DISABLE INTERRUPTS
 00120'066107           DOBS    1.MCAR
 00121'063607           SKPDN   MCAR
 00122'000777           JMP     .-1
 00123'060207           NIOC    MCAR    ;CLEAR DONE
 00124'060177           INTEN           ;ALLOW BASIC TO INTERRUPT
 00125'034475           LDA     3.CIAD  ;SET UP ADDRESS
 00126'054435           STA     3.IAI   ;FOR FLOATING IMAGS
 00127'054435           STA     3.OAI
 00130'030471           LDA     2.INPAD ;SET UP ADDRESS
 00131'050425           STA     2.IAR   ;FOR FLOATING
 00132'050425           STA     2.OAR   ;REALS
 00133'024020           LDA     1.20    ;SAVE AUTO-INC LOC
 00134'050020           STA     2.20
 00135'020721           LDA     0.NLR
 00136'040465           STA     0.CNT
 00137'101120           MOVZL   0.0
 00140'143000           ADD     2.0     ;ADDR PF BLOCK EXP
 00141'040417           STA     0.EXP1
 00142'040423           STA     0.EXP2
 00143'022020 MVLP:     LDA     0.@20   ;GET IMAGS
 00144'041400           STA     0.0.3   ;STORE ADJACENT IMAG ARRAY
 00145'022020           LDA     0.@20   ;GET NEXT REAL
 00146'041001           STA     0.1.2   ;COMPACT REALS
 00147'151400           INC     2.2
 00150'175400           INC     3.3
 00151'014452           DSZ     CNT
 00152'000771           JMP     MVLP
 00153'044020           STA     1.20
 00154'006453           JSR     @BFLT   ;FLOAT REALS
 00155'000056'          NLR
 00156'000000 IAR:      0
 00157'000000 OAR:      0
 00160'000000 EXP1:     0
 00161'006446           JSR     @BFLT   ;FLOAT IMAGS
 00162'000056'          NLR
 00163'000000 IAI:      0
 00164'000000 OAI:      0
 00165'000000 EXP2:     0
 00166'020670           LDA     0.NLR
 00167'101120           MOVZL   0.0
 00170'024431           LDA     1.INPAD ;INCREMENT INPUT (REAL)
 00171'107000           ADD     0.1     ;AND IMAGINARY BUFFER
 00172'044427           STA     1.INPAD ;ADDRESSES BY 2*NLR
 00173'024427           LDA     1.CIAD  ;FOR NEXT CHANNEL
 00174'107000           ADD     0.1
 00175'044425           STA     1.CIAD
 00176'014414           DSZ     LCNT
 00177'000713           JMP     RCVLP
 00200'024417 BACK:     LDA     1.PFLG
 00201'125113           MOVL#   1.1.SNC ;EXTEND SIGN OF PFLG
 00202'102401           SUB     0.0.SKP
 00203'102000           ADC     0.0
 00204'006001$          JSR     @.FLOT  ;CONVERT PFLG TO F.P.
 00205'030410           LDA     2.PAD   ;AND RETURN
 00206'041000           STA     0.0.2
```

```
0004  RMDFE
 00207'045001           STA     1.1.2
 00210'034410           LDA     3.RTN1
 00211'001403           JMP     3.3
 00212'000000 LCNT:     0
 00213'000000 RA:       0
 00214'000000 IA:       0
 00215'000000 PAD:      0
 00216'000217'PFAD:     PFLG
 00217'000000 PFLG:     0
 00220'000000 RTN1:     0
 00221'000000 INPAD:    0
 00222'000000 CIAD:     0
 00223'000000 CNT:      0
 00224'000000 RTN2:     0
 00225'000000 RTN3:     0
 00226'000000 ADR:      0
 00227'177777 BFLT:     BFLTM
                        .END
```

```
---
0001   BFLT
                ;9/3/75 COOKSON & SHIVELY
                ;SUBROUTINE TO CONVERT NORMALIZED BLOCK FLOATING POINT NUMBERS
                ;TO NORMALIZED DATA GENERAL FORMAT. THE BLOCK EXPONENT IS A
                ;POWER OF 2. MAY BE + OR -. -256<EXPONENT<256. AND THE MANTISSAE
                ;ARE SINGLE PRECISION 2'S COMPLEMENT FRACTIONS. B15. THIS PROGRAM
                ;HAS 2 ENTRY POINTS. ONE FOR CONVERTING A SINGLE NUMBER AND ONE
                ;FOR CONVERTING AN ARRAY. CALLING SEQUENCES ARE AS FOLLOWS:
                ;TO CONVERT A SINGLE NUMBER:
                ;       JSR     BFLTS
                ;       RETURN
                ;PARAMETERS:    AC0=THE MANTISSA
                ;               AC2=THE BLOCK EXPONENT
                ;RESULTS:       AC0=SIGN. EXPONENT & 8 MSB OF MANTISSA
                ;               AC1=16 LSB OF MANTISSA

                ;TO CONVERT AN ARRAY:
                ;       JSR     BFLTM
                ;ARAS:  0               ;ADDRESS OF ARRAY SIZE (NUMBER OF MANTISSA)
                ;INA:   0               ;ADDRESS OF INPUT ARRAY
                ;OUTA:  0               ;ADDRESS OF OUTPUT ARRAY
                ;EXPA:  0               ;ADDRESS OF BLOCK EXPONENT
                ;       RETURN
                ;THE ARRAY CONVERSION MAY BE DONE "IN PLACE" WHEN THE OUTPUT
                ;ARRAY ADDRESS IS SPECIFIED AS THE SAME AS THE INPUT. IN THIS
                ;CASE THE OUTPUT WILL EXTEND TO TWICE THE SIZE OF THE INPUT
                ;BEGINNING AT THE SAME ADDRESS.


                        .TITL   BFLT
                        .NREL
                        .ENT    BFLTS, BFLTM

00000'031400 BFLTM:  LDA     2,0,3   ;GET ADDRESS OF ARRAY SIZE
00001'021000         LDA     0,0,2   ;GET ARRAY SIZE
00002'024030         LDA     1,.INA  ;SAVE AUTO DECREMENT LOC 30 & 31
00003'044532         STA     1,AD30
00004'024031         LDA     1,.OUTA
00005'044531         STA     1,AD31
00006'040523         STA     0,COUNT
00007'175400         INC     3,3     ;TO NEXT PARAMETER
00010'031400         LDA     2,0,3   ;GET ADDRESS OF INPUT ARRAY
00011'113000         ADD     0,2     ;TO END OF ARRAY
00012'050030         STA     2,.INA  ;TO ALLOW WORK IN PLACE
00013'175400         INC     3,3     ;NEX PARAM
00014'031400         LDA     2,0,3   ;ADD OF OUTPUT ARRAY
00015'101120         MOVZL   0,0     ;OUTPUT SIZE=2*INPUT
00016'113000         ADD     0,2
00017'050031         STA     2,.OUTA ;TO OUTPUT BUFFER
00020'175400         INC     3,3     ;NEXT PARAM
00021'031400         LDA     2,0,3   ;GET ADDRESS OF BLK EXPONENT
00022'031000         LDA     2,0,2   ;GET EXPONENT
00023'175400         INC     3,3     ;TO RETURN
00024'054501         STA     3,EXFL

                ;FIND DATA GENERAL EXPONENT AND REMAINDER

00025'126400         SUB     1,1     ;GENERATE +2
00026'125520         INCZL   1,1
00027'176400         SUB     3,3     ;ZERO 3 TO RECEIVE REMAINDER
```

```
0002  BFLT
 00030'151113          MOVL#   2.2.SNC :CHECK SIGN OF 2 EXPO
 00031'000456          JMP     PEXB    :ITS +
 00032'150622          NEGZR   2.2.SZC :ITS -. TAKE MAG. DIV BY 2. CK FOR REM
 00033'175400          INC     3.3     :REM=1. ADD 1 TO SHIFT COUNT
 00034'151222          MOVZR   2.2.SZC :DIV BY 2 AGAIN. CK REM
 00035'137000          ADD     1.3     :REM=2. ADD 2 TO SHIFT
 00036'054475          STA     3.LRAS  :REM=NO. RT SHIFT
 00037'034471          LDA     3.C64   :CONVERT EXPO TO EXCESS 64
 00040'156700          SUBS    2.3     :INTO LEFT BYTE
 00041'171000          MOV     3.2     :SAVE IT IN 2
 00042'022030 CONV:    LDA     0.@.INA :GET MANTISSA
 00043'101122          MOVZL   0.0.SZC :CHECK SIGN
 00044'100401          NEG     0.0.SKP :TAKE ABS VALUE
 00045'176401          SUB     3.3.SKP :SET SIGN FLAG

 00046'176620          SUBZR   3.3
 00047'054463          STA     3.SIGN
 00050'034456          LDA     3.LBMSK :ARRANGE MANTISSA IN
 00051'105300          MOVS    0.1     :AC0 & AC1 IN DG FORMAT
 00052'163700          ANDS    3.0     :AC0=8 MSB
 00053'167400          AND     3.1     :8 MSB OF AC1=8 LSB OF MANT
 00054'034457          LDA     3.LRAS  :NO. RT SHIFT
 00055'174405          NEG     3.3.SNR :EXPO EVENLY DIVIS BY 4?
 00056'000405          JMP     NS      :YES. NO SHIFT
 00057'101220 SL:      MOVZR   0.0     :NO
 00060'125200          MOVR    1.1
 00061'175404          INC     3.3.SZR :COUNT UP
 00062'000775          JMP     SL
 00063'143000 NS:      ADD     2.0     :ADD IN EXCESS 64 EXPO
 00064'034446          LDA     3.SIGN  :SET SIGN
 00065'163000          ADD     3.0     :GOT DG FORMAT
 00066'034451          LDA     3.TAD
 00067'041400          STA     0.0.3
 00070'045401          STA     1.1.3
 00071'076374          DOBP    3.FPU1  :LOAD FPAC
 00072'060175          NIOS    FPU2    :NORMALIZE
 00073'076174          DOBS    3.FPU1  :STORE FPAC
 00074'021400          LDA     0.0.3
 00075'025401          LDA     1.1.3
 00076'046031          STA     1.@.OUTA:FULL BUFF FROM END
 00077'042031          STA     0.@.OUTA:AUTO DEC
 00100'014431          DSZ     COUNT   :DONE?
 00101'000741          JMP     CONV    :NO
 00102'020433          LDA     0.AD30  :YES. RESTORE AUTO DEC
 00103'040030          STA     0..INA
 00104'020432          LDA     0.AD31
 00105'040031          STA     0..OUTA
 00106'002417          JMP     @EXFL

               :FIND DG EXPO & REMAINDER FOR + BLK EXPO
 00107'020420 PEXB:    LDA     0.C4
 00110'151222          MOVZR   2.2.SZC :DIV BY 2. CK REM
 00111'175400          INC     3.3     :REM=1. 4-1 SHIFT
 00112'151222          MOVZR   2.2.SZC :DIV BY 2 AGAIN. CK REM
 00113'137000          ADD     1.3     :REM=2
 00114'175005          MOV     3.3.SNR :CK FOR 0 REM
 00115'102401          SUB     0.0.SKP :0 REM. EXACT DIV BY 4
 00116'151400          INC     2.2     :ADD 1 TO DG EXPO
 00117'162400          SUB     3.0     :4-REM=NO. SHIFT
```

```
0003  BFLT
 00120'040413           STA     0.LRAS
 00121'034407           LDA     3.C64    :CONVERT TO EXCESS 64
 00122'157300           ADDS    2.3      :INTO LEFT BYTE
 00123'171000           MOV     3.2      :SAVE IT IN 2
 00124'000716           JMP     CONV
 00125'000000 EXFL:     0
 00126'177400 LBMSK:    177400   :LEFT BYTE MASK
 00127'000004 C4:       4
 00130'000100 C64:      64.
 00131'000000 COUNT:    0
 00132'000000 SIGN:     0        :SIGN FLAG
 00133'000000 LRAS:     0        :NO. SHIFT
 00134'000000 TEMP:     0
 00135'000000 AD30:     0        :TEMP STOR AUTO DEC
 00136'000000 AD31:     0
 00137'000140'TAD:      TEM
       000002 TEM:      .BLK 2
       000074 FPU1=74
       000075 FPU2=75
       000030 .INA=30
       000031 .OUTA=31

              ;SINGLE NUMBER CONVERSION ROUTINE
 00142'054763 BFLTS:    STA     3.EXFL   :SAVE RETURN
 00143'101122           MOVZL   0.0.SZC  :CHECK SIGN
 00144'100401           NEG     0.0.SKP  :TAKE ABS VALUE
 00145'176401           SUB     3.3.SKP  :SET SIGN FLAG
 00146'176620           SUBZR   3.3
 00147'054763           STA     3.SIGN
 00150'034756           LDA     3.LBMSK  :ARRANGE MANTISSA IN
 00151'105300           MOVS    0.1      :DG FORMAT IN AC0&1
 00152'163700           ANDS    3.0
 00153'167400           AND     3.1
 00154'151113           MOVL#   2.2.SNC  :CK SIGN OF 2 EXPO
 00155'000431           JMP     PEX      :+
 00156'150623           NEGZR   2.2.SNC  :DIV BY 2. CK REM
 00157'000403           JMP     D10      :REM=0. NO SHIFT
 00160'101220           MOVZR   0.0
 00161'125200           MOVR    1.1
 00162'151223 D10:      MOVZR   2.2.SNC  :DIV BY 2. CK REM
 00163'000405           JMP     D20      :REM=0
 00164'101220           MOVZR   0.0      :REM=2. 2SHIFTS
 00165'125200           MOVR    1.1
 00166'101220           MOVZR   0.0
 00167'125200           MOVR    1.1
 00170'034740 D20:      LDA     3.C64    :CONV TO EXCESS 64
 00171'156700           SUBS    2.3      :SUB 2 EXPO/4 FROM 64
 00172'163000           ADD     3.0      :INSERT EXPO
 00173'034737 NRM:      LDA     3.SIGN   :FIX SIGN
 00174'163000           ADD     3.0      :DONE
 00175'034742           LDA     3.TAD
 00176'041400           STA     0.0.3
 00177'045401           STA     1.1.3
 00200'076374           DOBP    3.FPU1   :LOAD FPAC
 00201'060175           NIOS    FPU2     :NORMALIZE
 00202'076174           DOBS    3.FPU1   :STORE FPAC
 00203'021400           LDA     0.0.3
 00204'025401           LDA     1.1.3
 00205'002720           JMP     @EXFL
```

```
0004  BFLT

                ;FOR POSITIVE 2 EXPO
00206'176400 PEX:     SUB     3.3       ;SHIFTS=4-REM
00207'151222          MOVZR   2.2.SZC   ;DIV BY 2. CK REM
00210'175400          INC     3.3       ;NEED 3 OR LESS RT SHFT
00211'151223          MOVZR   2.2.SNC
00212'000403          JMP     R1        ;REM=0 THIS DIV
00213'175400          INC     3.3
00214'175400          INC     3.3
00215'175005 R1:      MOV     3.3.SNR   ;CK FOR EVEN DIV BY 4
00216'000413          JMP     EV4       ;EVEN 4. NO SHIFTS
00217'151400          INC     2.2       ;NEED TO SHFT. ADD 1 TO 16 EXPO
00220'040714          STA     0.TEMP    ;NEED AN AC
00221'020706          LDA     0.C4
00222'162400          SUB     3.0       ;SH=4-REM
00223'114400          NEG     0.3       ;SHIFT COUNT
00224'020710          LDA     0.TEMP    ;RECALL MSB
00225'101220 ISL:     MOVZR   0.0       ;SHIFT MANTISSA
00226'125200          MOVR    1.1
00227'175404          INC     3.3.SZR
00230'000775          JMP     ISL
00231'034677 EV4:     LDA     3.C64     ;CONVERT TO EXCESS 64
00232'173300          ADDS    3.2       ;INTO L BYTE
00233'143000          ADD     2.0       ;PUT IN EXPO
00234'000737          JMP     NRM
                      .END
```

```
0001  RLOAD

                        .TITL   RLOAD
                ;       9/17/75         C. SHIVELY (R.H. DAVIS)
                ;                       & J.P. COOKSON
                ;       PROGRAM TO TRANSMIT VIA THE MCA A SAVE FILE
                ;       FROM A NOVA HAVING DOS SUPPORT TO THE MEMORY
                ;       OF ANOTHER NOVA FOR EXECUTION
                ;       THE RECEIVING MACHINE SHOULD BE "PROGRAM
                ;LOADED" WITH BIT 0 SET ON AND DEVICE CODE
                ;       07 (MCAR) IN THE CONSOLE SWITCHES
                ;       COMMAND FORMAT:
                ;       RLOAD/S1 FILE.SV/S2/S3
                ;       S1      MCA RECEIVER CODE
                ;       A       10
                ;       B       04
                ;       C       02
                ;       D       01
                ;       S2 OR S3        LOCATION/START MODE
                ;       S                       AFTER LOADING. START PROGRAM
                ;                               EXECUTION AT CONTENTS OF LOC 405
                ;                               (HALT AFTER LOADING IF NOT USED)
                ;       Z                       START LOADING SAVE FILE IN LOC 0
                ;                               (LOAD AT LOC 16 IF NOT USED)
                ;       IF THE DESIRED SAVE FILE DOES NOT HAVE THE
                ;       SV NAME EXTENSION. SIMPLY GIVE THE FILE NAME
                        .NREL
00000'020547 RLOAD:     LDA     0,.CFNBP ;COM.CM BYTE POINTER
00001'126400            SUB     1,1
00002'006002            .SYSTM
00003'074001            .OPEN   1       ;OPEN COM.CM ON 1
00004'000465            JMP     ERROR
00005'020543            LDA     0,.PRNBP ;PROGRAM NAME BYTE POINTER
00006'006002            .SYSTM
00007'075401            .RDL    1       ;READ OVER "RLOAD"
00010'000461            JMP     ERROR
00011'020541            LDA     0,.DSBP  ;DSW BYTE POINTER
00012'024520            LDA     1,C4
00013'006002            .SYSTM
00014'075001            .RDS    1 .     ;GET DEVICE SWITCH WORD
00015'000454            JMP     ERROR
00016'020533            LDA     0,.IFNBP
00017'006002            .SYSTM
00020'075401            .RDL    1       ;GET INPUT FILENAME
00021'000465            JMP     INARG   ;ERROR EXIT
00022'020531            LDA     0,.MSBP  ;MODE SW BYTE POINTER
00023'024507            LDA     1,C4    ;GET SAVE FILE INITIAL LOC
00024'006002            .SYSTM          ;AND START/HALT SWITCHES
00025'075001            .RDS    1
00026'000443            JMP     ERROR
00027'006002            .SYSTM
00030'074401            .CLOSE  1       ;DONE READING COM.CM
00031'000440            JMP     ERROR
00032'020517 OPNIN:     LDA     0,.IFNBP
00033'126400            SUB     1,1
00034'006002            .SYSTM
00035'074001            .OPEN   1       ;OPEN INPUT FILE ON 1
00036'000433            JMP     ERROR
00037'030537            LDA     2,.MSW+1 ;CHECK FOR S AND Z SWITCHES
00040'034501            LDA     3,.SMSK  ;OCTAL 020000 MASK
```

```
0002  RLOAD
 00041'157404           AND     2.3.SZR ;IF S SWITCH SET. EXECUTE
 00042'000403           JMP     .+3     ;SAVE FILE AFTER LOADING
 00043'020430           LDA     0.HLT   ;OTHERWISE. SET UP
 00044'042477           STA     0.@.STI ;BOOTSTRAP TO HALT
 00045'034475           LDA     3.ZMSK  ;OCTAL 000100 MASK
 00046'157404           AND     2.3.SZR ;IF Z SWITCH SET. LOAD
 00047'000403           JMP     .+3     ;SAVE FILE STARTING AT LOC 0
 00050'020464           LDA     0.C16   ;OTHERWISE. SET UP BOOTSTRAP
 00051'042473           STA     0.@.SAD ;TO START LOAD AT LOC 16
 00052'020473           LDA     0.BBLKP ;BOOTSTRAP BLOCK POINTER
 00053'024464           LDA     1.C256  ;SEND 256 WORD BOOTSTRAP
 00054'004437           JSR     SEND    ;TO SLAVE CPU
 00055'020477 RDLUP:    LDA     0.OBKBP ;READ NEXT 256
 00056'024462           LDA     1.C512  ;WORDS FROM INPUT FILE
 00057'006002           .SYSTM
 00060'075001           .RDS    1
 00061'000405           JMP     CKEOF   ;IF ERROR CHECK FOR EOF
 00062'020464           LDA     0.OBLKP ;SEND NEXT 256
 00063'024454           LDA     1.C256  ;WORDS TO SLAVE CPU
 00064'004427           JSR     SEND
 00065'000770           JMP     RDLUP
 00066'020445 CKEOF:    LDA     0.C6
 00067'112415           SUB#    0.2.SNR ;IF EOF ERROR.
 00070'000404           JMP     ENDFL   ;SEND LAST BLOCK
 00071'006002 ERROR:    .SYSTM          ;ERROR RETURN TO DOS
 00072'066400           .ERTN           ;WITH ERROR MESSAGE
 00073'063077 HLT:      HALT            ;ON TTO
 00074'020452 ENDFL:    LDA     0.OBLKP
 00075'024442           LDA     1.C256  ;SEND LAST BLOCK OF
 00076'125400           INC     1.1     ;257 WORDS TO SLAVE CPU
 00077'004414           JSR     SEND
 00100'006002           .SYSTM
 00101'074401           .CLOSE  1       ;CLOSE INPUT FILE ON 1
 00102'000767           JMP     ERROR
 00103'006002           .SYSTM
 00104'064400           .RTN            ;RETURN TO DOS
 00105'063077           HALT
 00106'020425 INARG:    LDA     0.C6    ;IF EOF ERROR.
 00107'112414           SUB#    0.2.SZR ;NO INPUT FILE NAME GIVEN
 00110'000761           JMP     ERROR   ;OTHER ERROR EXIT
 00111'030425           LDA     2.CB100 ;IF NOT INPUT FILE NAME
 00112'000757           JMP     ERROR   ;TYPE "NOT ENOUGH ARGUMENTS"
 00113'060277 SEND:     INTDS           ;GET FREE OF DOS
 00114'124400           NEG     1.1
 00115'066206           DOBC    1.MCAT  ;SET COUNT
 00116'061006           DOA     0.MCAT  ;SET ADDRESS
 00117'024454           LDA     1.DSW   ;SELECT RECEIVER
 00120'067106 SEND1:    DOCS    1.MCAT  ;START TRANSFER
 00121'063606           SKPDN   MCAT
 00122'000777           JMP     .-1
 00123'062606           DICC    0.MCAT  ;CHECK STATUS
 00124'060177           INTEN           ;ALLOW CTRL A INTERRUPT
 00125'101200           MOVR    0.0     ;IF TRANSMITTER WORD
 00126'101202           MOVR    0.0.SZC ;COUNT DONE. RETURN
 00127'001400           JMP     0.3     ;FOR NEXT BLOCK
 00130'060277           INTDS           ;OTHERWISE. TRY TO FORM
 00131'000767           JMP     SEND1   ;LINK AND SEND AGAIN
      000001            .TXTM   1
 00132'000004 C4:       4
```

```
0003  RLOAD
 00133'000006 C6:      6
 00134'000016 C16:     16
 00135'000031 CB31:    31
 00136'000100 CB100:   100
 00137'000400 C256:    256.
 00140'001000 C512:    512.
 00141'020000 3MSK:    020000
 00142'000100 ZMSK:    000100
 00143'000262'.STI:    STI
 00144'000244'.SAD:    SAD
 00145'000211'BBLKP:   BTSTP
 00146'000611'OBLKP:   OBLK
 00147'000332"CFNBP:   2*CFNME
 00150'000342"PRNBP:   2*PRNBK
 00151'000376"IFNBP:   2*IFNME
 00152'000366"DSBP:    2*DSW
 00153'000372"MSBP:    2*MSW
 00154'001422"OBKBP:   2*OBLK
       000001          .TXTM   1
               CFNME:  .TXT    *COM.CM*
 00155'041517
 00156'046456
 00157'041515
 00160'000000
       000012 PRNBK:   .BLK    10.
       000002 DSW:     .BLK    2
       000002 MSW:     .BLK    2
       000012 IFNME:   .BLK    10.
              ;        9/17/75          J.P. COOKSON & C.SHIVELY
              ;THE FOLLOWING PROGRAM IS DESIGNED TO BE LOADED INTO
              ;MEMORY LOCATIONS 0 THROUGH 377 OCTAL OF
              ;A MACHINE VIA THE DATA CHANNEL WHILE THAT MACHINE IS
              ;EXECUTING A JMP 377 IN 377. IT FIRST SIZES MEMORY THEN
              ;TRANSFERS A BLOCK OF SIZE "BSIZE" BEGINNING IN LOCATION
              ;"LBTX" TO UPPER MEMORY. FROM "THE LAST LOC - BSIZE + 1"
              ;THRU THE LAST LOCATION. IT THEN JUMPS TO THE FIRST LOC
              ;WHICH WAS TRANSFERRED TO UPPER MEMORY.
              ;THE INTENTION IS TO USE THIS PROGRAM AS A "BOOTSTRAP"
              ;TO BE LOADED VIA THE MCA WITH THE ASSISTANCE OF THE
              ;INTERNAL ROM BOOTSTRAP AND SOME TRANSMITTER UNDER DOS.
              ;THE BLOCK TRANSFERRED TO UPPER MEMORY IS TO BE ANOTHER
              ;PROGRAM CAPABLE OF LOADING A SAVE FILE VIA THE MCA


 00211'102000 BTSTP:   ADC     0,0     ;-1 TO AC0
 00212'034433          LDA     3,K2    ;THE TEST LOC PTR IN 3 IS 1024
 00213'165000          MOV     3,1     ;POINTER TO 1
 00214'030432          LDA     2,C31   ;MAX MEM=31K
 00215'050433          STA     2,COUNT ;SET UP COUNTER
 00216'041400 MEML:    STA     0,0,3   ;STORE -1
 00217'031400          LDA     2,0,3   ;LOAD IT BACK
 00220'112404          SUB     0,2,SZR ;DID IT COME BACK?
 00221'000404          JMP     TEND    ;NO. FOUND THE END
 00222'137000          ADD     1,3     ;YES. GO UP 1024 & TRY AGAIN
 00223'014425          DSZ     COUNT   ;CHECK UP TO 31K
 00224'000772          JMP     MEML    ;NO. CONTINUE LOOKING
 00225'020424 TEND:    LDA     0,BSIZE ;GET SIZE OF REST OF BOOT
 00226'040422          STA     0,COUNT ;FOR XFER TO UPPER MEM
 00227'116400          SUB     0,3     ;1ST LOC IN UPPER BOOT
```

```
0004  RLOAD
 00230'054422            STA     3..UBT  ;SAVE IT TO GO THERE
 00231'030422            LDA     2..LBTX ;1ST LOC IN LOWER BOOT FOR XFER
 00232'025000 BTXF:      LDA     1.0.2   ;DO THE XFER
 00233'045400            STA     1.0.3
 00234'151400            INC     2.2
 00235'175400            INC     3.3     ;NEXT INST FOR XFER
 00236'014412            DSZ     COUNT   ;FINISHED?
 00237'000773            JMP     BTXF.   ;NO. XFER NEXT INSTRUCTION
 00240'020404            LDA     0.SAD   ;SET RECEIVER START ADDRESS
 00241'061007            DOA     0.MCAR
 00242'024405            LDA     1.M257  ;SET RECVR WD COUNT = 257
 00243'002407            JMP     @.UBT   ;YES. GO TO UPPER BOOT
 00244'000000 SAD:       0               ;SAVE FILE 1ST LOCATION
 00245'002000 K2:        2000            ;1024 = MIN MEM INCREMENT
 00246'000037 C31:       31.             ;MAX ADDRESS IS 31K
 00247'177377 M257:      -257.
 00250'000000 COUNT:     0
 00251'000010 BSIZE:     10              ;SIZE OF XFER TO UPPER MEM
 00252'000000 .UBT:      0               ;TO BEGIN OF UPPER BOOT
 00253'000043 .LBTX:     43              ;TO BEGIN OF LOWER BOOT XFER
 00254'066107 LBTX:      DOBS    1.MCAR  ;SET RECVR WD COUNT = 257
 00255'063607            SKPDN   MCAR    ;WAIT UNTIL DONE SET
 00256'000777            JMP     .-1
 00257'062407            DIC     0.MCAR  ;IF 257 WORDS RECEIVED.
 00260'101203            MOVR    0.0.SNC ;LAST BLOCK. OTHERWISE.
 00261'000773            JMP     LBTX    ;RECEIVE ANOTHER BLOCK
 00262'002401 STI:       JMP     @.+1    ;JMP THROUGH LOC 405
 00263'100405            100405          ;TO PROGRAM STARTING ADDRESS
       000324            .BLK    324     ;GAP TO PUT NEXT
 00610'000000            JMP     0       ;INSTRUCTION IN LOC 377
       000401 OBLK:      .BLK    257.
         000000'           .END    RLOAD
```

NSWC/WOL TR 77-3

Name: RLOAD

Format: RLOAD SAVE_FILENAME

Purpose: To transmit via the MCA a save file from the disk on a Nova with DOS to the memory of another Nova for execution

Switches:

Global: Required to specify the destination computer. The following four letter codes are recognized:

/A computer with MCA code 1000 (8)

/B computer with MCA code 0100 (4)

/C computer with MCA code 0010 (2)

/D computer with MCA code 0001 (1)

Local: /S - Start save file execution after loading by jumping to the starting address set by the relocatable loader in location 405 of the save file. If not used, the destination computer will halt in location HCM-1 after loading, where HCM is the last location in the lowest contiguous memory block.

/Z - Begin loading save file at location Zero in the destination computer. (The save file must have been created using the /Z switch with the RLDR command.) If not used, loading of the save file will start in location 16_8.

Asterisk: Not permitted.

Errors: ILLEGAL FILE NAME

FILE DOES NOT EXIST

NOT ENOUGH ARGUMENTS

Examples: RLOAD/C PROG.SV
causes save file PROG.SV to be loaded into computer C beginning in location 16_8 with halt after loading.

RLOAD/D FOO/Z/S
causes save file FOO to be loaded into computer D beginning in location 0 with automatic transfer to FOO's start address.

NSWC/WOL TR 77-3

Note: The destination computer should be executing a JMP 377 in location 377, with the MCA receiver set up to receive at least 256 words starting in location 0. If the MCA receiver has been modified so that an IORST resets its word and address counters to 0, simply program load with bit 0 set on and device code 07 (MCAR) in the console switches. Otherwise, the following program must be executed in the destination computer:

```
START:  SUB     0,0     ;102400
        DOA     0,MCAR  ;061007
        DOBS    0,MCAR  ;062107
        LDA     0,C377  ;020402
        STA     0,377   ;040377
C377:   JMP     377     ;000377
```

The RLOAD command may be used to advantage for reloading DOS onto a disk from another DOS disk on which the system save file SYS.SV resides. For example, to restore DOS on disk C from disk B, issue the following commands:

```
RLOAD/C SYS.SV/Z/S      (on B)
    loads DOS into C core

LOAD/A/V $PTR           (on C)
    loads MCAR.SV onto C disk from dump tape

MCAT/C SYS.SV           (on B)
MCAR                    (on C)
    loads DOS save file onto C disk from B disk

INSTALL SYS.SV          (on C)
    installs DOS to boot from C disk
```

```
TYPE SPT.CS
0010 REM      PROGRAM TO INITIALIZE PAPB &PAPABFP
0020 REM      AND READ SPECTRUM. PRINTS LARGEST
0030 REM      POWER CELL
0040 REM      INPUT PARAMETERS FROM TERMINAL
0050 PRINT "NCH=",
0060 INPUT N
0065 PRINT
0070 PRINT "NL=",
0080 INPUT M
0085 PRINT
0090 PRINT "NLR=",
0100 INPUT L
0105 PRINT
0110 PRINT "RLD=",
0120 INPUT D
0125 PRINT
0130 DIM R[L*N],I[L*N],S[L]
0140 REM      INIT CPU A,B
0150 CALL 11,N,M,L,D
0160 REM      WAIT FOR DFT FROM A
0170 CALL 12,R[0],I[0],P
0180 IF P=0 THEN GOTO 0050
0190 REM      SUM SQUARES FOR POWER
0200 FOR J=0 TO L-1
0210  LET S[J]=R[J]^2+I[J]^2
0220 NEXT J
0230 REM      FIND CELL WITH LARGEST POWER
0240 LET B=0
0250 LET C=0
0260 FOR J=0 TO L-1
0270  IF S[J]<B THEN GOTO 0300
0280  LET B=S[J]
0290  LET C=J
0300 NEXT J
0310 PRINT C,B
0320 GOTO 0170
R
```

DISTRIBUTION

	Copies
Defense Documentation Center Cameron Station Alexandria, VA 22314	2

NDW-NSWC(W)-5605/1 (Rev. 1-75)

TO AID IN UPDATING THE DISTRIBUTION LIST FOR NAVAL SURFACE WEAPONS CENTER, WHITE OAK LABORATORY TECHNICAL REPORTS PLEASE COMPLETE THE FORM BELOW:

TO ALL HOLDERS OF NSWC/WOL TR 77-3
by Curtis A. Shively, Code CU-22
DO NOT RETURN THIS FORM IF ALL INFORMATION IS CURRENT

A. FACILITY NAME AND ADDRESS *(OLD) (Show Zip Code)*

NEW ADDRESS *(Show Zip Code)*

B. ATTENTION LINE ADDRESSES:

C.

☐ REMOVE THIS FACILITY FROM THE DISTRIBUTION LIST FOR TECHNICAL REPORTS ON THIS SUBJECT.

D.
NUMBER OF COPIES DESIRED _____

GPO 888-070

DEPARTMENT OF THE NAVY
NAVAL SURFACE WEAPONS CENTER
WHITE OAK, SILVER SPRING, MD. 20910

OFFICIAL BUSINESS
PENALTY FOR PRIVATE USE, $300

POSTAGE AND FEES PAID
DEPARTMENT OF THE NAVY
DOD 316

COMMANDER
NAVAL SURFACE WEAPONS CENTER
WHITE OAK, SILVER SPRING, MARYLAND 20910

ATTENTION: CODE CU-22